

FAST MULTISCALE FEATURE ANALYSIS VIA
OVERCOMPLETE WAVELET REPRESENTATIONS

By

MINBO SHIM

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULTILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1998

ACKNOWLEDGEMENTS

I would like to give my sincere thanks to my supervisor, Dr. Andrew F. Laine, chairman of my graduate supervisory committee, for his support during those years that I have been in the Ph.D. program at the Computer and Information Science and Engineering Department.

I would also thank Dr. Douglas K. Benn, cochairman of the supervisory committee, for his support, and his continuous concerns about my family. I would like to express my gratitude to Dr. Joseph N. Wilson for being a mentor to the computer vision in the first years of my research career. I am also grateful to Dr. Fred J. Taylor and Dr. John G. Harris for serving on my committee and for their comments for the completion of the dissertation.

I would like to give my special thanks to my father, Dr. Jae-Hoo Shim and my mother, Myung-Sun Park, for their endless support and encouragement.

Finally, I am very thankful to my wife, Seikyung, and my son, Michael, for their understanding and patience during the extent of my studies. Without their bearing with me when I was unbearable, I could have never completed my thesis at all. I am extremely thankful for their sincere love.

TABLE OF CONTENTS

	<u>page</u>
ACKNOWLEDGEMENTS	ii
LIST OF FIGURES	v
LIST OF TABLES	viii
ABSTRACT	ix
1 INTRODUCTION	1
2 MOTIVATIONS	3
3 MULTIREOLUTION ANALYSIS AND WAVELETS	13
3.1 Multiresolution Analysis	13
3.2 Wavelet Bases	16
3.3 Wavelet Transforms	22
4 COMPUTATIONAL COST OF DISCRETE WAVELET TRANSFORMS	24
4.1 Direct Filter Bank Algorithm	24
4.2 FFT-based DWT Algorithm	27
4.3 Short-length FIR Filtering	31
5 MULTISCALE BLOCK ALGORITHM	36
5.1 Architectural Overview of SPARCstation SX System	36
5.2 XIL Imaging Library	38
5.3 Multiscale Block Algorithm	40
6 LIFTING SCHEME	48
6.1 Second Generation Wavelets	48
6.2 Biorthogonal Wavelets and Duals	50
6.3 Lifting Scheme	55
6.4 Interpolating Scaling Functions, Associated Wavelets and Lifting Scheme	60
6.5 Generalized Lifting Scheme	68
6.6 Computational cost of the Lifting Scheme	75
7 OVERCOMPLETE LIFTING SCHEME	78

7.1 Undecimated Discrete Wavelet Transform with Lifting	79
7.2 Computational Cost of Overcomplete Lifting Scheme	84
8 CONCLUSIONS	95
REFERENCES	96
BIOGRAPHICAL SKETCH	99

LIST OF FIGURES

Figure	page
2-1 Example of wavelet coefficients showing boundary effects propagating through the continuous scale space.	5
2-2 Illustrations of boundary effects among different signal extension modes at the boundaries. (a) 1-D original signal with discontinuities in the middle. (b) Cropped signal showing the discontinuities in the original signal. (c) Wavelet coefficients from zero-padded extension. The wavelet coefficients clearly show the boundary effects. (d) Detail coefficients from symmetric extension. The detail coefficients show both the boundary effects and the discontinuities. (e) Wavelet coefficients from smooth padding. The coefficients show the discontinuities, but not the boundary effects.	6
2-3 Illustrations of boundary effects on an image based on zero-padding, symmetrization and smooth padding extensions. Level 5 decomposition is performed using Daubechies tap 5 wavelet, and then reconstruct the approximation of level 5. (a) Original image. (b) (c) (d) Reconstructed images using zero-padding, symmetrization and smooth padding extensions, respectively.	7
2-4 (a) Input 3-D volume from axial CT slices. (b) 3-D wavelet coefficients simulated with 2-D wavelet coefficients. (c) 3-D wavelet maxima edges obtained from 2-D multiscale edges.	10
2-5 3-D volume of sinuses extracted from the local maxima representation shown in Figure 2-4(c).	11
3-1 Time-frequency windows.	18
4-1 Basic computational cells for the forward DWT and the inverse DWT (IDWT).	24
4-2 Forward polyphase representation reorganized to allow fast filtering algorithms.	29
4-3 FFT-based implementation structure of the DWT cell shown in Figure 4-2.	30
4-4 Comparison of computational cost of the direct filter bank method, the FFT version of the DWT algorithm, and the fast running short-length FIR filtering.	33
4-5 An example of fast running FIR filtering algorithm for short-filters. Here decimation ratio is two.	34
5-1 SPARCstation SX system high-level architecture [26].	38
5-2 Solaris foundation graphics libraries and layered interface.	39

5-3	Difference between a linear convolution with zero-paddings in the kernel (a) and an alternative method with four sub-regions (b).	41
5-4	Multiresolution block convolution for image synthesis. Four sub-regions in the input image at a coarser level, marked as {I, II, III}, and {IV}, are merged into a larger region at a finer level. Then, convolution is applied to the merged region.	42
5-5	(a) Selected ROI containing an ill-defined mass. (b) DC components at levels $l = 1$, $l = 2$ and $l = 3$ after applying Algorithm 1. (c) Processed ROI with suspicious area enhanced.	45
5-6	Comparison of computational costs between multiscale block, traditional convolution and FFT-base approach.	47
6-1	A two channel subband coding scheme.	52
6-2	Graph showing the limiting process to make the Daubechies tap 4 scaling function. From top to bottom, scaling functions without iteration, after one iteration, after two iterations, and after fifteen iterations, respectively.	54
6-3	Graph of root mean square errors involved in the limiting steps to make the Daubechies's tap 4 scaling function shown in Figure 6-2.	55
6-4	A 1-D discrete wavelet transform with lifting polynomials.	58
6-5	Canonical structure in implementing 1-D discrete forward wavelet transform with lifting.	59
6-6	Wavelet representations of a ROI in a mammogram computed via lifting for three levels of decomposition.	61
6-7	Interpolating scaling functions resulting from interpolating subdivisions of order 2, 4, 6 and 8 from top to bottom	62
6-8	The cubic ($N = 4$) interpolating scaling functions near boundary changing their shapes to alleviate boundary effects.	63
6-9	Wavelets with two (dual) vanishing moments () associated with interpolating scaling functions whose orders of the subdivision are 2, 4, 6, and 8 from top to bottom.	66
6-10	Wavelets accommodating the left boundary. Here $N = 4$ and	67
6-11	Discrete wavelet transform in z-domain.	69
6-12	Polyphase representation of wavelet transform	70
6-13	Noble identities for multirate systems. Identity systems for (a) a decimator, and (b) an interpolator.	72
6-14	Forward wavelet transform using a dual polyphase matrix with finite lifting steps.	74
6-15	Inverse wavelet transform using the polyphase matrix with finite lifting steps.	75
6-16	Comparison of the costs between various DWT algorithms described in	

Chapter 4 and the lifting scheme.	77
7-1 Undecimated version of two channel filter bank with the proposed overcomplete lifting.	81
7-2 Forward wavelet transform to extract detail information as failures to be cubic.	81
7-3 Implementation of the overcomplete lifting scheme with multiscale block algorithm (a) Original signal with discontinuities near the middle. (b) Blocked input signal. (c) Blocked wavelet coefficients after lifting. (d) Unblocked coefficients.	83
7-4 Examples of 2-D masks of dual lifting coefficients for $N = 4$. (a) A mask for regions which are not affected by the boundaries. (b) A mask for the point (0, 0) of an input image. Rectangles indicate points that are modified by masking operations.	84
7-5 Wavelet coefficients of a mammogram with the proposed scheme. The first and second column show the wavelet coefficients in horizontal direction and vertical direction, respectively.	85
7-6 Costs of ODWT(Direct), ODWT(FFT-based), ODWT(Lifting) and ODWT (Liftingblock) algorithms with filter length (a) $L = 2$, (b) $L = 3$, (b) $L = 5$, and (d) $L = 7$. Here $n = 7$ ($N = 128$).	88
7-7 Same as Figure 7-6 with $n = 8$ ($N = 256$).	89
7-8 Same as Figure 7-6 with $n = 9$ ($N = 512$).	90
7-9 Costs of ODWT(direct), ODWT(FFT-based), ODWT(lifting) and ODWT (liftingblock) algorithms up to analyzing levels (a) $J = 3$, (b) $J = 4$, (b) $J = 5$, and (d) $J = 6$. Here $n = 7$ ($N = 128$).	91
7-10 Same as Figure 7-9 with $n = 8$ ($N = 256$).	92
7-11 Same as Figure 7-9 with $n = 8$ ($N = 512$).	93

LIST OF TABLES

<u>Table</u>	<u>page</u>
4-1 Computational costs per point and per elementary cell for the direct filter bank algorithm (4.2), the FFT-based algorithm (4.17) and the fast running short-length FIR filtering algorithm (4.19).	32
5-1 Computational costs in time (seconds) of splitting (in decomposition) and merging (in reconstruction) schemes versus traditional linear convolution with zero-padding between each non-zero filter coefficients and FFT implementations.	46
6-1 Lifting coefficients based on the number of on its left and right. (a), (b), (c) and (d) show the filter coefficients for $N = 2$, $N = 4$, $N = 6$ and $N = 8$, respectively	64

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

FAST MULTISCALE FEATURE ANALYSIS VIA
OVERCOMPLETE WAVELET REPRESENTATIONS

By

Minbo Shim

August 1998

Chairman: Andrew F. Laine

Cochairman: Douglas K. Benn

Major Department: Computer and Information Science and Engineering

The lifting scheme is a flexible tool for constructing compactly supported second generation wavelets which are not necessarily translates and dilates of one *mother* wavelet. The lifting scheme uses a simple basis function and builds a better performing one with desirable properties. Flexibility afforded by the lifting scheme allows basis functions to change their shapes near the boundaries without degrading regularities. The lifting scheme also provides fast processing by making optimal use of similarities between high and low pass filters. The lifting algorithm was originally introduced to construct biorthogonal wavelets associated with interpolating scaling functions, and was generalized later using a finite number of lifting steps by means of the Euclidean

algorithm.

The lifting scheme utilizes a classical 2-channel filter bank as a framework for multiresolution analysis. However this traditional framework is not translation invariant. Representations with a translation-invariant characteristic are highly desirable for feature analysis. In this dissertation we address the following question: Can the lifting scheme be used as a framework for overcomplete wavelet representations with multiscale feature analysis in mind? Our work addresses this question by investigating each stage of the multiscale analysis: Split, Dual lifting and Primal lifting. We introduce a *smoothing Lazy* wavelet in the *split* stage which does not subsample, but smooths an input image so that the low-pass channel contains some redundant information. Since the new *split* operation does not use downsampling to make distinct subsets, there is no aliasing. We show that the primal lifting required to compensate for the aliasing by preserving energy between two contiguous approximations needs no longer exist. Only the *predict* stage, based upon specific properties, indeed leads to a useful multiresolution analysis which is applicable to multiresolution feature analysis. We also show in our work that the proposed scheme achieves better performance without introducing boundary artifacts that exist in the traditional methods. This makes the overcomplete lifted wavelet transform very viable in interactive processing paradigms such as a Web-based client-server model for multiscale feature analysis where fast processing is highly desirable.

CHAPTER 1

INTRODUCTION

Wavelet functions $\psi_{j,m}$ are traditionally defined as the translates and dilates of one particular function in $L_2(\mathbf{R})$, *mother wavelet* ψ , such that $\psi_{j,m}(x) = \psi(2^j x - m)$. Although these wavelets have a nice property of forming a Riesz basis for $L_2(\mathbf{R})$, discrete implementations typically require compromised solutions such as zero padding, symmetric extension, or periodization to overcome artifacts around boundaries. Furthermore, the translation invariance desirable in multiscale algorithms for multiscale feature analysis is not preserved in the standard wavelet transform as described by Mallat [19]. In addition, the standard algorithm in the filter bank structure is not necessarily the best way to implement a wavelet transform. There have been several schemes to improve the standard fast wavelet transform such as split-radix FFT algorithms and short-length FIR filtering. For long length filters, an FFT based scheme known as Vetterli-algorithm was suggested, and for short length filters, a fast running FIR algorithm was formulated by Mou and Duhamel [21].

Recently, the concept of *lifting* was introduced by Wim Sweldens [29] [30] to alleviate the problems associated with a traditional digital filter bank scheme. In this scheme, basis functions associated with each wavelet coefficient are no longer necessarily dilates and translates of a *mother* wavelet. Flexibility afforded by the lifting scheme allows the basis functions near boundaries to change their general shape for better accommodation. Also the lifting scheme allows a faster processing by making optimal use

of similarities between high and low pass filters to speed up the calculation of coefficients. Ingrid Daubecheies and Wim Sweldens [11] also showed that the lifting is general in that every wavelets and wavelet transform with finite filters can be obtained with a finite number of lifting steps.

In this dissertation, a comprehensive study on the construction of wavelets and their associated transform using the lifting scheme shall be thoroughly carried out starting from a biorthogonal wavelet construction and constructions of orthogonal and semiorthogonal wavelets with a finite number of lifting steps. We show during the study how the boundary artifacts existing in a standard wavelet transform via a filter bank structure are taken care of with interpolating lifting schemes. A new algorithm, *Overcomplete Lifting Scheme (OLS)*, for overcomplete wavelet representations is formulated as a mathematical problem. Error to signal ratio for the reconstructions is computed to determine robustness of the new framework. Finally computational complexity analysis and comparisons are carried out to identify the speedup the overcomplete lifting scheme can achieve. From the comparison we show that the *OLS* with a block algorithm is most efficient.

CHAPTER 2 MOTIVATIONS

In this chapter we describe practical backgrounds that have motivated our research. We first discuss boundary effects resulting from various extensions to handle a finite length input. We then show that overcomplete wavelet representations have an increased amount of data to be represented and processed. Finally, we describe a demand of high speed transform for interactive image processing.

Wavelet theory provides a systematic framework for multiresolution analysis. Wavelets are versatile tools for representing functions or data sets in a multiscale space. The capability of time-frequency localization of wavelets makes representations of signals efficient [10]. The justification of using dyadic wavelets and wavelet transform has been supported by several preliminary results [1] [2] [3] [16] [17].

Several algorithms were reviewed by Rioul and Duhamel [23] for computing various types of wavelet transforms. Mallat [19] introduced a fast discrete dyadic wavelet transform via a filter bank structure: the *standard wavelet transform*, where four filters are needed to compute forward and inverse processing steps. While the filters do not change during the computation, the computation of the filter coefficients is not a trivial task. In addition, the standard wavelet transform suffers from boundary artifacts if the length of input is not power of two, a common problem associated with digital filters operating on a finite sets of data. There have been numerous attempts at minimizing this boundary effect. The most common approaches have been zero-padding, symmetric extension,

periodization and smooth extension. These workaround approaches are effective in theory but are not entirely satisfactory from a practical viewpoint. Figure 2-1 shows an idea about the signal end effects. In this figure, the wavelet coefficients not only capture the location in time of the discontinuities, but also display the evolution of unwanted boundary artifacts as the scale grows. Figure 2-2 illustrates the boundary effects generated by some signal extensions: zero-padding, symmetrization and smooth padding. Zero-padding assumes that the input signal is zero outside the support. This extension obviously introduces artificial discontinuities at the border. Symmetrization, which works if the basis functions are symmetric, is to use reflection across edges. This extension mode preserves continuity, but also introduces discontinuities in the first derivative. Coefficients created by these two extension modes shown in Figure 2-2(c) and Figure 2-2(d) respectively clearly demonstrate that wavelets designed for detecting singularities within finite-length signals likely consider the boundaries as other singularities of the input signals. The third extension method shown in Figure 2-2(e), smooth padding, extends the finite support of the input signal by a first order derivative extrapolation. Figure 2-3 illustrates the boundary effects introduced by the extensions described applied to an image. While the end effects based upon smooth padding extension did not appear in 1-D signal case (see Figure 2-2(e)), some artifacts clearly appear in 2-D image case (see Figure 2-3(d)). Thus we need basis functions that overcome the undesirable effects near boundaries without any extension.

The standard dyadic wavelet transform given by

$$[W_{2^j} f(2^j n) = \langle f(x), \tilde{\psi}_{2^j}(x - 2^j n) \rangle]_{(n,j) \in \mathbb{Z}^2} \quad (2.1)$$

provides discrete, uncorrelated and complete representation of the function $f(x)$. One of

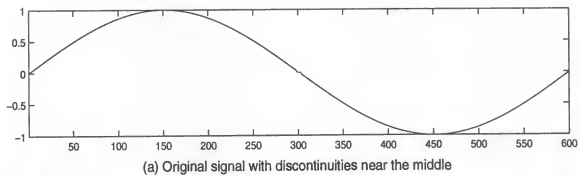


Figure 2-1. Example of wavelet coefficients showing boundary effects propagating through the continuous scale space.

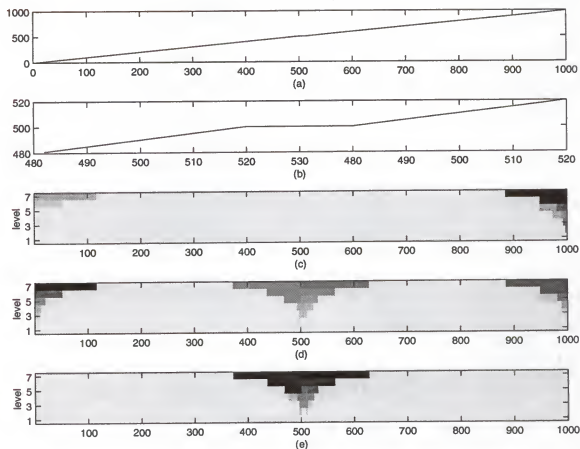


Figure 2-2. Illustrations of boundary effects among different signal extension modes at the boundaries. (a) 1-D original signal with discontinuities in the middle. (b) Cropped signal showing the discontinuities in the original signal. (c) Wavelet coefficients from zero-padded extension. The wavelet coefficients clearly show the boundary effects. (d) Detail coefficients from symmetric extension. The detail coefficients show both the boundary effects and the discontinuities. (e) Wavelet coefficients from smooth padding. The coefficients show the discontinuities, but not the boundary effects.

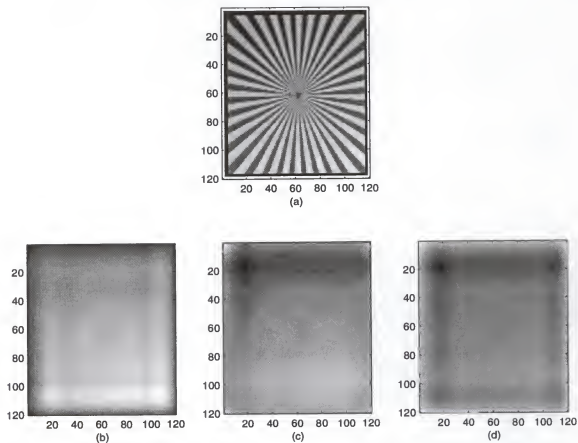


Figure 2-3. Illustrations of boundary effects on an image based on zero-padding, symmetrization and smooth padding extensions. Level 5 decomposition is performed using Daubechies tap 5 wavelet, and then reconstruct the approximation of level 5. (a) Original image. (b) (c) (d) Reconstructed images using zero-padding, symmetrization and smooth padding extensions, respectively.

the main advantages of wavelet transform over Fourier transform is the time-frequency localization. The wavelet transform based upon some basis function ψ has a zoom-in and zoom-out capability in time-frequency space. In other words, the wavelet transform has a flexible time-frequency window that automatically narrows at high center frequency and widens at low center frequency.

However, such representations may not be well suited to pattern recognition or feature analysis because of its non-invariance with translations [20]. In order to preserve translation invariance, each function $W_2 f(x)$ is oversampled to give a redundant representation. This representation however needs considerably increased amount of data for processing. An adaptive sampling using local wavelet maxima has been proposed by Mallat and Zhong [20]. Since the multiscale edge representations are sparse compared to original data, the processing cost based on these representations may be reduced.

Benn et al. [1] used the local maxima representations to extract volumes of interest for automated knowledge-based feature recognition. We developed a novel method to represent multiscale objects with maxima surface: *Isomaxima* surface.

We first apply a 1-D discrete dyadic wavelet transform in x, y and z directions to a CT volume data set to obtain analyzing coefficients (gradient vectors). Then we compute modulus and two directions from the coefficients that detect boundaries of objects by locating a local maximum in the direction of the gradient. The maxima are then triangulated based on certain properties, such as a (local or global) regularity of the surface, directional constraints and a magnitude tolerance between nearby maxima. Arbitrary triangular meshes are converted to a multiresolution form that has *subdivision connectivity*. The multiresolution mesh representations finalized can be used in

compression, incremental display and transmission, level-of-detail control and multiresolution editing.

3-D wavelet modulus and angle are given respectively by

$$M_{jf} = \sqrt{|W_j^1 f|^2 + |W_j^2 f|^2 + |W_j^3 f|^2},$$

$$A_{jf} = \{\arctan(\alpha), \arctan(\beta)\} = \left\{ \arctan\left(\frac{W_j^2 f}{W_j^1 f}\right), \arctan\left(\frac{W_j^3 f}{W_j^1 f}\right) \right\}.$$

The 3-D maxima is then defined by

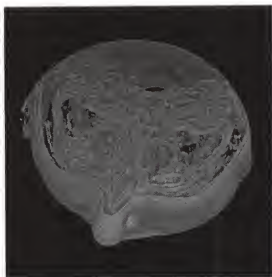
$$\hat{M}_{jf} = \begin{cases} M_{jf}, & \text{if } M_{jf} \text{ is maximum along the gradient direction } A_{jf}, \\ 0, & \text{Otherwise.} \end{cases}$$

Then a maxima vertex v_n at level j can be found if it satisfies the following condition: $|\hat{M}_{jf}(v_n) - \hat{M}_{jf}(v_m)| < T^\alpha$, where v_m is a neighboring maxima vertex, T is a threshold and α is a regularity constant. The isomaxima surface would then be used in multiresolution surface analysis proposed by Eck *et al.* [14] and Lounsbery *et al.* [18]. Figure 2-4(a) shows an input 3-D volume from axial CT slices. Figure 2-4(b) and 2-4(c) show simulated output of a 3-D wavelet representation and a corresponding 3-D multiscale edges, respectively. A 3-D segmentation of maxillary sinuses is shown in Figure 2-5.

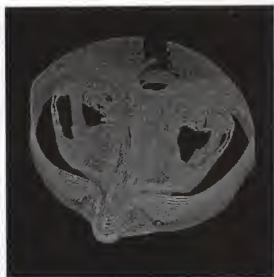
However, in order to construct 3-D volumes of interest from one study of a patient, we need approximately 140MB ~ 1500MB disk space. The numbers come from the following specifications: image size is 512 by 512, pixel depth 2 bytes, number of slices in one study 30 ~ 120, 3-way (x , y and z directional) transform, and level of analysis practically goes from 3 to 8. The wavelet transform on the 3-D data set is thus very computationally burdensome and required huge amount of auxiliary memory. A wavelet



(a)



(b)



(c)

Figure 2-4. (a) Input 3-D volume from axial CT slices. (b) 3-D wavelet coefficients simulated with 2-D wavelet coefficients. (c) 3-D wavelet maxima edges obtained from 2-D multiscale edges.

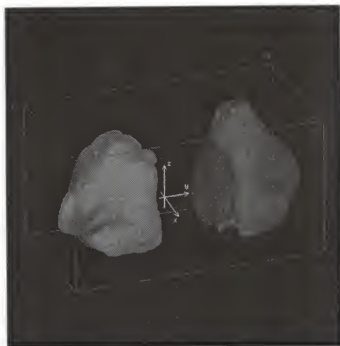


Figure 2-5. 3-D volume of sinuses extracted from the local maxima representation shown in Figure 2-4(c).

transform with an in-place implementation is demanding.

The increased amount of data for overcomplete representations where an FFT has been the basic implementation tool are also computationally burdensome and inefficient. As pointed out by Rioul and Duhamel [23], the FFT method performs best when the filter length is long (over 16 taps). Since the filters we are using in the transform are compact, we may have a faster transform by avoiding the FFT. Practically, a faster wavelet transform is needed for applications where real time interaction is required. We designed a Web page (<http://www.iprg.cise.ufl.edu/rsna97/>) originally created for RSNA (Radiological Society of North America) 1997 exhibition in Chicago, Illinois. The Web page was designed such that radiologists and medical imaging professionals could learn about and use the multiscale techniques for mammographic feature enhancement developed by Laine *et al.* [17] and Iztok *et al.* [16]. A faster algorithm was demanded to accomplish high-speed interactive multiscale processing for the wavelet-based enhancement. Otherwise, the embedded nature of real-time interaction of the Web-based documenting would lose its virtue.

CHAPTER 3 MULTIRESOLUTION ANALYSIS AND WAVELETS

In this chapter, we briefly describe multiresolution analysis, wavelets and wavelet transforms that are needed to understand and situate the remainder of the dissertation. We first review general properties of wavelet and various wavelet transforms which have become known as useful tools for various signal or image processing applications. The definitions of multiresolution analysis and scaling function are given in the next section, where we give the basic definition of wavelet as well. We also study the connection between wavelets and polynomials, and show how this relates to the approximation properties of wavelet expansions.

3.1 Multiresolution Analysis

A multiresolution analysis of $L^2(R)$ is defined as a sequence of closed subspaces $V_j \subset L^2(R)$, $j \in Z$, with the following properties:

- 1) $V_j \subset V_{j+1}$,
- 2) for all $j \in Z$, $f(x) \in V_{2j} \Leftrightarrow f(2x) \in V_{2j+1}$,
- 3) for all $l \in Z$, $f(x) \in V_0 \Leftrightarrow f(x+l) \in V_0$,
- 4) $\lim_{j \rightarrow \infty} V_{2j} = \bigcup_{j=-\infty}^{\infty} V_{2j}$ is dense in $L^2(R)$,
- 5) $\lim_{j \rightarrow -\infty} V_{2j} = \bigcap_{j=-\infty}^{\infty} V_{2j} = \{0\}$,

- 6) a scaling function $\psi \in V_0$, with a non-vanishing integral, exists so that the wavelet family $\{\psi_{j,k} | k \in \mathbb{Z}\}$ forms a Riesz basis of V_0 .

The first property is a causality property. The information of a signal present at a scale $j+1$ contains all necessary information to generate the same signal at a coarser level j . The properties (2) and (3) are scale and shift invariance, respectively. In other words, the property (2) tells us that the operation of multiresolution analysis is unique at all resolutions, so that each space of approximated functions in $L^2(\mathbb{R})$ should be able to be derived from one another. The property (3) is an isomorphism. When $f(x)$ is translated by a length proportional to 2^j , the approximation of $f(x)$ is translated by the same amount and is characterized by the same samples which have been translated. The properties (4) and (5) are about the convergence. When computing an approximation of $f(x)$ at resolution 2^j , some information about $f(x)$ is lost. Property (4) tells that as the resolution increases to $+\infty$, the limiting signal should converge to the original signal. Property (5) is the converse of property (4), that is, as the resolution decreases, the approximation contains less and less information and converges to zero.

Property (1) says that if $\varphi(x)$ is in V_0 , then it is also in V_1 . The scaling function thus satisfies

$$\varphi(x) = 2 \sum_k h_k \varphi(2x - k) . \quad (3.1)$$

This is the *dilation equation*. It is a *two-scale difference equation*, involving x and $2x$. It is also called a *refinement equation*, because it displays $\varphi(x)$ in the refined space V_1 . We refer to h as the filter associated with the scaling function φ . Under general conditions, the scaling function is uniquely defined by the refinement equation and the normalization,

such that

$$\int_{-\infty}^{\infty} \varphi(x) dx = 1 .$$

This implies that $\sum_k \varphi(x-k) = 1$, so that as given in property (6), the integer translates of the scaling function $\{\varphi(x-l) | l \in \mathbb{Z}\}$ form a Riesz basis for the closure of their span, and also partition unity. The refinement equation (3.1) is equivalent to

$$\hat{\varphi}(\omega) = h\left(\frac{\omega}{2}\right) \hat{\varphi}\left(\frac{\omega}{2}\right),$$

where $h(\omega)$ is defined as

$$h(\omega) = \sum_k h_k e^{-ik\omega} .$$

Since $\hat{\varphi}(0) = 1$, iteration of the refinement relation given in (3.1) yields the product formula for the scaling function as

$$\hat{\varphi}(\omega) = \prod_{j=1}^{\infty} h(2^{-j}\omega), \quad (3.2)$$

which can be used to construct $\varphi(x)$ from the sequence $\{h_k\}$.

Multiresolution analysis can also be achieved by building a multiresolution representation based on the difference of information available between two contiguous resolutions 2^j and 2^{j+1} . The difference of information is called the *detailed signal* at resolution 2^j which is given by the orthogonal projection of the original signal on the orthogonal complement space of V_j in V_{j+1} . Let O_j denote this orthogonal complement space. Then it satisfies

$$V_{j+1} = V_j \oplus O_j .$$

Since the space O_j contains the detail information needed to generate an approximation at

resolution $j + 1$, it also satisfies

$$\bigoplus_j O_j = L^2(R) .$$

A wavelet $\psi(x)$ is an orthonormal basis of O_j needed to compute the orthogonal projection of a function $f(x)$ on O_j . The collection of wavelets $\{\psi(x-l)|l \in \mathbb{Z}\}$ forms a Riesz basis of $L^2(R)$ as well. The wavelet ψ also satisfies the refinement relation

$$\psi(x) = 2 \sum_k g_k \psi(2x-k) , \quad (3.3)$$

where we refer to g as the filter associated with ψ . The refinement equation is equivalent to

$$\hat{\psi}(\omega) = g\left(\frac{\omega}{2}\right) \hat{\psi}\left(\frac{\omega}{2}\right) ,$$

$$\text{where } g(\omega) = \sum_k g_k e^{-ik\omega} .$$

3.2 Wavelet Bases

We described in the previous section that the basis functions for the space V^j are called scaling functions. We defined the wavelet spaces, denoted by W^j , where detail information between two contiguous spaces is projected as an orthogonal complement of V^j in V^{j+1} . Thus, any function in V^{j+1} can be written as the sum of a function in V^j and a function in W^j . The functions we choose as a basis for W^j are called wavelets. In this section, we review general properties of the wavelet.

Wavelet transforms compute coefficients that are inner products of a time-varying signal $x(t)$ and a family of wavelets

$$\psi_{a,b}(t) = |a|^{-1/2} \psi\left(\frac{t-b}{a}\right) , \quad (3.4)$$

where $\psi(t)$ is the prototypical analyzing wavelet, or called a *mother* wavelet, a represents a scale factor in time, and b a time location. The factor $|a|^{-1/2}$ is used for energy preservation. The *admissibility* condition is required on $\psi(t)$ for it to be considered as an analyzing wavelet [10], which is given as

$$C_\psi = \int \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty, \quad (3.5)$$

where $\hat{\psi}(\omega)$ is the Fourier transform of $\psi(t)$. This condition ensures that the wavelet transform is a bounded and invertible operator in $L^2(R)$. In particular, if $\psi(t)$ must be a window function, then $\psi(t)$ is necessarily in $L^1(R)$, meaning $\int |\psi(t)| dt < \infty$, so that $\hat{\psi}(\omega)$ is differentiable. Hence from (3.5) it must vanish at the origin: that is,

$$\int \psi(t) dt = 0. \quad (3.6)$$

In order for (3.6) to be satisfied, $\psi(t)$ must be oscillated. Since $\hat{\psi}(0) = 0$ and the power spectrum $|\hat{\psi}(\omega)|$ decays at high frequencies, admissible wavelets often act the way that band-pass filters do.

There is an outstanding property in the wavelet basis: *time-frequency* localization with varying window shape due to different time and frequency intervals, which makes the wavelet and wavelet transform more viable than the short-time Fourier transform (STFT) which uses a constant time-frequency window. The variation of resolution in the time-frequency phase space enables the wavelet transform to zoom into the irregularities of the signal and characterize them locally. To explain this property, let x_o and ω_o be finite centers, and Δ_x and Δ_ω finite radii of a wavelet ψ , respectively. Then they are defined as

$$x_o = \frac{1}{\|\psi\|^2} \int_{-\infty}^{\infty} x |\psi(x)| dx ,$$

$$\Delta_x^2 = \frac{1}{\|\psi\|^2} \int_{-\infty}^{\infty} (x - x_o)^2 |\psi(x)|^2 dx ,$$

and

$$\omega_o = \frac{1}{\|\hat{\psi}\|^2} \int_{-\infty}^{\infty} \omega |\hat{\psi}(\omega)| d\omega ,$$

$$\Delta_\omega^2 = \frac{1}{\|\hat{\psi}\|^2} \int_{-\infty}^{\infty} (\omega - \omega_o)^2 |\hat{\psi}(\omega)|^2 d\omega .$$

The energy of the continuous wavelet transform defined in (3.13) is mostly distributed in a time-frequency window defined by the time interval $[b + ax_o - a\Delta_x, b + ax_o + a\Delta_x]$ and the frequency interval $\frac{1}{a}[\omega_o - \Delta_\omega, \omega_o + \Delta_\omega]$. The importance of the time-frequency window is that it widens for small center-frequency, and narrows for large center-frequency $\frac{\omega_o}{a}$. This is shown in Figure 3.4. However the area of the time-frequency window is constant and given by $4\Delta_x\Delta_\omega$. This is exactly what is most desirable in time-frequency analysis. The constant windowing area suggests that the product of the time

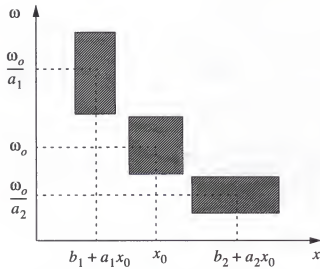


Figure 3-1. Time-frequency windows.

interval and the frequency interval is a stable quantity. The Heisenberg Uncertainty Principle sets a specific lower bound for the product, which is given as

$$\Delta_x \Delta_\omega > 1 .$$

From the admissibility condition given in (3.5) and $\hat{\psi}(0) = 0$, the frequency localization of wavelets is much better than described above. Since ω_o is zero in most cases, the frequency interval is further divided into a subband $\frac{1}{a}\{[-\omega_2, -\omega_1] \cup [\omega_2, \omega_1]\}$

There are three categories of wavelet bases: orthogonal, semiorthogonal and biorthogonal. An orthogonal basis is one in which each basis function is orthogonal to every other basis function. An orthogonal multiresolution basis is one in which the scaling functions are orthogonal to one another, the wavelets are orthogonal to one another, and each of the wavelets is orthogonal to every scaling function. Wavelets that satisfy these conditions are called orthogonal wavelets. These conditions can be equivalently written for every j, k and l as

$$\begin{aligned} \langle \phi_k^j | \phi_l^j \rangle &= \delta_{k,l} , \\ \langle \psi_k^j | \psi_l^j \rangle &= \delta_{k,l} , \\ \langle \phi_k^j | \psi_l^j \rangle &= 0 , \end{aligned} \tag{3.7}$$

where $j, k, l \in Z$. The Haar wavelet and the Daubechies wavelets are the orthogonal wavelets.

Orthogonality is not the only desirable property in a wavelet basis. There are some other properties that may be more important when constructing a wavelet for a particular application: compact support, symmetry, smoothness and vanishing moments. A more compact support improves the efficiency of decomposing and reconstructing functions using a filter bank algorithm. Smooth functions are best represented with smooth bases.

However, greater smoothness comes at the expense of wider supports. Wavelets with more vanishing moments are often desirable for applications that require numerical approximations of smooth operators. It is known that there are no wavelet bases except the Haar wavelet that are orthogonal, compactly supported and symmetric at the same time [9] [10]. For those applications that need to use smooth symmetric wavelets with compact supports, orthogonality may be sacrificed. Wavelets that are orthogonal to all scaling functions, not necessarily orthogonal to each other are called semiorthogonal wavelets. Thus the only condition required of the semiorthogonal wavelets is $\langle \phi_k^j | \psi_l^j \rangle = 0$ for all j, k and l . Spline-based wavelets are the semiorthogonal wavelets.

Semiorthogonal constructions however take quadratic time in analysis although reconstruction can be done in linear time. Cohen, *et al.* [7] developed biorthogonal wavelets that although not orthogonal to scaling functions, still have many of the desirable properties of the semiorthogonal wavelets. Dual basis functions are central to the definition of a biorthogonal wavelet construction. Let $\mathbf{u}(x)$ be a set of orthonormal bases such that $\mathbf{u}(x) = [u_1(x) u_2(x) \dots]$. Then a function $f(x)$ can be represented with coefficients c_i in the span of $\mathbf{u}(x)$:

$$f(x) = \sum_i c_i u_i(x). \quad (3.8)$$

A coefficient c_j can be determined by the inner product of the function with $u_j(x)$:

$$\begin{aligned} \langle f | u_j \rangle &= \sum_i c_i \langle u_j | u_i \rangle \\ &= \sum_i c_i \delta_{i,j} \\ &= c_j. \end{aligned} \quad (3.9)$$

If $\mathbf{u}(x)$ is not an orthonormal basis, then $\langle f | u_j \rangle \neq c_j$. However we can find a set

of bases $\tilde{u}(x) = [\tilde{u}_1(x) \tilde{u}_2(x) \dots]$ that computes the coefficient c_j :

$$c_j = \langle f | \tilde{u}_j \rangle. \quad (3.10)$$

Such functions $\tilde{u}(x)$ are called the dual basis. By taking the inner product of both sides of (3.10), we see that

$$\langle u_j | \tilde{u}_j \rangle = \delta_{i,j}. \quad (3.11)$$

A biorthogonal wavelet basis is one in which the primal scaling functions are orthogonal to the dual wavelets and the primal wavelets are orthogonal to the dual scaling functions. In mathematical notation, the conditions defining a biorthogonal wavelet basis can be written for every j, k and l as

$$\begin{aligned} \langle \phi_k^j | \tilde{\phi}_l^j \rangle &= \delta_{k,l} \\ \langle \psi_k^j | \tilde{\psi}_l^j \rangle &= \delta_{k,l} \\ \langle \phi_k^j | \tilde{\psi}_l^j \rangle &= 0 \\ \langle \psi_k^j | \tilde{\phi}_l^j \rangle &= 0. \end{aligned} \quad (3.12)$$

Deciding whether to use an orthogonal, semiorthogonal or biorthogonal wavelets is not a trivial task, since each has its own advantages and disadvantages. One of advantages of orthogonal wavelets is that L^2 norm can be easily computed from the coefficients. Semiorthogonal wavelets are useful in that they provide compactly supported and symmetric bases while maintaining orthogonality between wavelets and scaling functions. Biorthogonal wavelets are the most flexible ones among all. Lifting that will be discussed in detail later provides a flexible tool to construct the biorthogonal wavelets by improving such properties as locality of support and degree of orthogonality.

3.3 Wavelet Transforms

Based on the type of the parameter pair (a, b) in Equation (3.4), we produce a different type of wavelet transform: continuous wavelet transform (CWT), wavelet series transform (WST) and discrete wavelet transform (DWT). In the CWT, time t and the time-scale parameter pair (a, b) vary continuously, which is given as

$$CWT\{x(t), (a, b)\} = \int \bar{\psi}_{a,b}(t)x(t)dt, \quad (3.13)$$

where $\bar{\psi}(t)$ is complex conjugate of $\psi(t)$, and $b \in R, a > 0$. In WST, a has the form $a = 2^j$ where j is termed the octave of the transform, and b a multiple of a : that is, $b = n2^j$. Here, $j, k \in Z$. The WST is thus given as

$$WST\{x(t), (2^j, k2^j)\} = 2^{-j/2} \int \bar{\psi}(2^j t - n)x(t)dt. \quad (3.14)$$

In DWT, both time and time-scale parameters are discrete. The DWT is thus given as

$$DWT\{x[n], (2^j, k2^j)\} = 2^{-j/2} \sum_k \bar{\psi}(2^j k - n)x[k]. \quad (3.15)$$

The sample rate has been set to one. From the implementation point of view, the DWT is in fact the same as an octave-band filter bank which is computationally efficient. As pointed out by Rioul and Duhamel [23], if a wavelet transform is the DWT, then the implementation is thus likely to be efficient and fast. The arithmetic complexity of various DWT algorithms will be shown in the next chapter.

A function can be reconstructed from its wavelet coefficients. With the constant C_ψ satisfying (3.5), we have the following reconstruction formula:

$$f(x) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W_\psi f(a, b) \psi_{a,b}(a, b) \frac{da db}{a^2}, \quad (3.16)$$

where $W_\psi f(a, b)$ is a set of wavelet coefficients. Since the CWT can only be computed on

a discrete grid of points, we focus on use of the DWT for recovery of the function. Since only partial information of $W_\psi f$ is available in the DWT, the basis wavelet ψ must satisfy a stronger condition than the admissibility condition (3.5) for a reconstruction formula to be available. One of the important properties to be considered in this case is its completeness or stability. In order to establish stable representations, the basis must form a Riesz basis satisfying

$$A\|f\|^2 \leq \sum |\langle f, \psi_{a,b} \rangle|^2 \leq B\|f\|^2, \quad (3.17)$$

where A and B , with $0 < A \leq B < \infty$, are constants independent of ω , and both a and b are dyadic numbers. The wavelet family $\{\psi_{a,b} | a, b \in Z\}$ constitutes a *frame* if it satisfies (3.17). If $A = B$, then the frame is called a *tight frame*. Thus, for any $f \in L^2(R)$, we have

$$f = \sum_{a,b \in Z} \langle f, \tilde{\psi}_{a,b} \rangle \psi_{a,b},$$

where $\tilde{\psi}$ is a dual of ψ . Here if $A = B = 1$ and $\tilde{\psi}_{a,b} = \psi_{a,b}$, then the wavelet family $\psi_{a,b}$ constitutes an orthonormal basis.

CHAPTER 4 COMPUTATIONAL COST OF DISCRETE WAVELET TRANSFORMS

The discrete wavelet transform (DWT) has been known as a natural wavelet transform for discrete-time signals. In this chapter, we study computational cost of various fast algorithms to implement the DWT. We define the computational cost as the sum of the number of multiplications and the additions.

4.1 Direct Filter Bank Algorithm

Among several implementation structures for the DWT, the octave-band filter bank shown in Figure 4-1 has been recognized as a general and regular structure where basic computational cells for the forward and inverse DWT are repeated without any

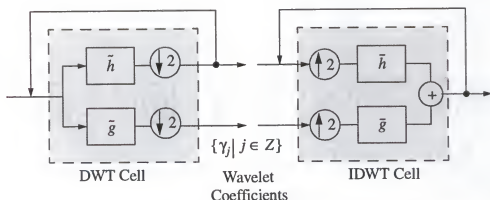


Figure 4-1. Basic computational cells for the forward DWT and the inverse DWT (IDWT).

modification throughout the DWT implementation. Although this general method is already efficient, several authors [21] [22] [23] have shown that there is a room for noticeable computational savings by FFT-based fast convolution techniques.

Before calculating computational cost of efficient algorithms, we assume for simplicity that signal and finite-length filters are real. This assumption applies in general, such that it can be extended to the computation of the cost for the complex-valued case for which we only need to know additional multiplications and additions required. For straightforward filter bank algorithms, each complex multiplication requires four real multiplications and two real additions, and each complex addition requires two real additions.

One other assumption to make is that the filters involved in the computation of the DWT have equal length. This is true for the orthogonal DWT, while in the biorthogonal case the filter lengths may differ. However the equal length restriction can hold for the biorthogonal case as well by padding shorter filters with zeros to have the same length as the longest filter.

An algorithm performs faster and is more efficient than others if there is some reduction of computational cost for the algorithm. Here the cost is usually derived based on the number of real multiplications and real additions per input point required by the algorithm. With today's technology, multiplications are no longer dominant part of the computations since a parallel multiplier is built in the modern computers. Thus a useful criterion is the sum of the number of multiplications and additions. Although it is not the only relevant criterion, the computational cost is fairly instructive way for comparing various DWT algorithms.

The DWT implementation on the filter bank is known fast. An intuitive reason is that the DWT uses the decomposition of the computation into basic computational cells and the decimations within the cell. Let us suppose that the filters, $h[n]$ and $g[n]$, have equal length L . Then a point in the DWT cell requires $2L$ multiplications and $2(L-1)$ additions for every set of two inputs. If we let M_j^{direct} and A_j^{direct} be the number of real multiplications and additions per input point for each computational cell at the j th octave, respectively, then for $j = 1, 2, \dots, J$

$$\begin{aligned} M_j^{direct} &= L, \\ A_j^{direct} &= L - 1. \end{aligned} \quad (4.1)$$

That is, the number of operations per input point for each computational elementary cell is given by

$$nOPS_{direct} = (L \oplus (L - 1)), \quad (4.2)$$

where $nOPS_{direct}$ stands for the number of operations involved in the direct filter bank, and $(m \oplus a)$ denotes m multiplications and a additions per a point for an elementary cell.

The input to the DWT cell at the j th octave is downsampled by 2^{j-1} . Thus the total number of input points involved in the DWT on J octaves is $2N(1 - 2^{-J})$, where N is the size of the original input. If we denote by C_{direct} the total cost of the standard DWT algorithm on J octaves implemented directly as a filter bank, then

$$C_{direct} = 2N(1 - 2^{-J})(L \oplus L - 1). \quad (4.3)$$

As the number of octaves, J , increases, C_{direct} is bounded to $2N(L \oplus (L - 1))$. Since the filter length is the major factor in comparing DWT algorithms, we consider N as a constant. Thus

$$O(C_{direct}) \approx L . \quad (4.4)$$

The DWT (3.15) can be written in terms of an associated filter $g_j[n]$ by

$$DWT\{x[n], (2^j, k2^j)\} = \sum_n x[n] \bar{g}_j[n - 2^j k] . \quad (4.5)$$

In the naive computation of DWT (4.5), the length of the filter $g_j[n]$ at the j th octave is $(L - 1)(2^j - 1) + 1$. Thus

$$\begin{aligned} M_j^{direct} &= (L - 1)(2^j - 1) + 1 , \\ A_j^{direct} &= (L - 1)(2^j - 1) . \end{aligned} \quad (4.6)$$

Let C_{naive} be the total cost of the naive DWT algorithm on J octaves. Then

$$C_{naive} = ((JN(L - 1) + 1) \oplus (JN(L - 1))) . \quad (4.7)$$

Thus

$$O(C_{naive}) \approx JL . \quad (4.8)$$

The complexity (4.8) does not bound as the J increases. Based on the complexities given by (4.4) and (4.8) as big O notations, the filter bank algorithm implementing the DWT achieves a speedup by a factor of J over the naive computation of the DWT algorithm by reducing the complexity from JL to L .

4.2 FFT-based DWT Algorithm

Although the filter bank algorithm described in the previous section already achieves a fast implementation, FFT-based algorithms lead to further reduction of the computational complexity of the DWT from L to $\log L$. In this section we introduce the *split-radix* FFT algorithm [22] which, among all practical FFT algorithms, has the best known complexity for lengths $N = 2^n$ [23]. Before going into more details on the

algorithm, we need to reorganize the filter bank structure shown in Figure 4-1 to be more efficient. Since each computational cell contains two filters followed by subsampling, a polyphase representation is better in terms of efficiency. Filtering in the polyphase implementation structure is applied to each of the even and odd subsampled sequence separately. This already reduces the computation by a factor of two.

The polyphase representation of x is given by

$$X(z) = X_e(z^2) + z^{-1}X_o(z^2), \quad (4.9)$$

where X_e and X_o contain even and odd sequence, respectively. In other words,

$$\begin{aligned} X_e(z) &= \sum_k x_{2k} z^{-k}, \\ X_o(z) &= \sum_k x_{2k+1} z^{-k}. \end{aligned} \quad (4.10)$$

Let us denote by $A(z)$ the cell output that enters the next stage, and by $B(z)$ the cell output that computes wavelet coefficients at the current stage. $A(z)$ (respectively, $B(z)$) is obtained by filtering by $H(z)$ (respectively, $G(z)$) followed by subsampling, that is

$$\begin{aligned} A(z) &= H_e(z)X_e(z) + z^{-1}G_o(z)X_o(z), \\ B(z) &= G_e(z)X_e(z) + z^{-1}H_o(z)X_o(z). \end{aligned} \quad (4.11)$$

Figure 4-2 shows the polyphase structure for the forward DWT based on (4.11) that reorganizes the DWT cell shown in Figure 4-1.

Many FFT algorithms based on decimation-in-time and decimation-in-frequency use sequences whose length is power of two. There have been FFT algorithms that make the discrete Fourier transform efficient if N can be represented as a product of factors. The *radix-R* FFT algorithm is one of them that uses a product of an identical factor (i.e., $N = R^v$). The *radix-2* and *radix-4* algorithms are the ones that have been mostly used for

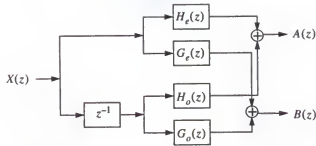


Figure 4-2. Forward polyphase representation reorganized to allow fast filtering algorithms.

practical applications. The *split-radix* FFT algorithm [22] combines them in such a way that it requires the lowest number of multiplications and additions and keeps the same regularity as the radix-4 algorithm and the same flexibility as the radix-2 algorithm. It uses a fact that a radix-4 is better for odd terms of the DFT and a radix-2 for the even terms of the DFT. If we define the complex quantity in the DFT as $W_N = e^{-j(2\pi/N)}$, the split-radix is decomposed into

$$\begin{aligned}
 X_{2k} &= \sum_{n=0}^{N/2-1} (x_n + x_{n+(N/2)}) W_N^{2nk}, \\
 X_{4k+1} &= \sum_{n=0}^{N/4-1} [(x_n - x_{n+(N/2)}) - j(x_{n+(N/4)} - x_{n+(3N/4)})] W_N^n W_N^{4nk}, \\
 X_{4k+3} &= \sum_{n=0}^{N/4-1} [(x_n - x_{n+(N/2)}) + j(x_{n+(N/4)} - x_{n+(3N/4)})] W_N^{3n} W_N^{4nk}.
 \end{aligned} \tag{4.12}$$

The split-radix decimation-in-frequency decomposition replaces a DFT of length N by one DFT of length $N/2$ and two DFT's of length $N/4$. Let ${}_R M_N^{FFT}$ and ${}_R A_N^{FFT}$ be the number of real multiplications and additions, respectively, needed to perform a 2^n -real

DFT with the split-radix algorithm. Then the split-radix FFT for real data requires [22]
[23]

$$\begin{aligned} R M_n^{FFT} &= 2^{n-1}(n-3) + 2, \\ R A_n^{FFT} &= 2^{n-1}(3n-5) + 4. \end{aligned} \quad (4.13)$$

For complex data,

$$\begin{aligned} C M_n^{FFT} &= 2^n(n-3) + 4, \\ C A_n^{FFT} &= 3 \cdot 2^n(n-1) + 4, \end{aligned} \quad (4.14)$$

where $C M_n^{FFT}$ and $C A_n^{FFT}$ are the number of complex multiplications and additions.

In order to compute the computational cost of FFT-based DWT algorithms, Figure 4-3 shows an alternative implementation structure of the DWT cell depicted in Figure 4-2. The input of size N is split into even and odd sequences, and a length $\frac{N}{2}$ FFT is applied to each sequence separately. The four convolutions with four filters in frequency domain are followed. This requires four $\frac{N}{2}$ complex multiplications. Two blocks are then added which requires two $\frac{N}{2}$ complex additions, and finally two $\frac{N}{2}$ inverse FFT's are applied. A complex multiplication requires four real multiplications and two real additions. The total

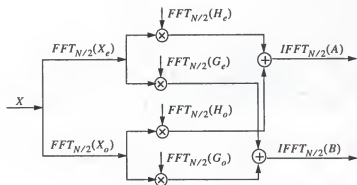


Figure 4-3. FFT-based implementation structure of the DWT cell shown in Figure 4-2.

number of operations for an input of size N involved in the FFT-based implementation structure shown in Figure 4-3 are thus given as

$$2 \cdot FFT_{N/2} + 4 \cdot (4mults + 2adds) \cdot \frac{N}{2} + 2 \cdot 2 \cdot \frac{N}{2} adds + 2 \cdot IFFT_{N/2}. \quad (4.15)$$

Here we assume that the FFT's of the four filters are precomputed. For general purpose, we also assume that all inputs for the FFT and IFFT are complex, so that we use the cost on complex data (4.14). The total number of operations per input point for each basic computational cell is thus given as

$$\left(\left(\frac{2^{n+1}n + 16}{2^n} \right) \oplus \left(\frac{3 \cdot 2^{n+1}(n-1) + 16}{2^n} \right) \right). \quad (4.16)$$

Since L has the same order of magnitude as N , $n = \log L$. If we substitute every n in (4.16) with $\log L$, the number of operations per input point in the FFT-based computational cell $nOPs_{FFT}$ is given by

$$nOPs_{FFT} = \left(\left(\frac{2L \log_2 L + 16}{L} \right) \oplus \left(\frac{6L(\log_2 L - 1) + 16}{L} \right) \right). \quad (4.17)$$

Table 4-1 shows the resulting computational costs for different lengths L in comparison with the direct method (4.2) and with the FFT-based algorithm (4.17). The comparison tells us that the FFT version of the DWT algorithm is efficient for filters of length greater than or equal to 12 ($L \geq 12$). Figure 4-4 shows graphically the costs of $nOPs_{direct}$ and $nOPs_{FFT}$ based on the results in Table 4-1.

4.3 Short-length FIR Filtering

Since the FFT-based algorithms are no longer efficient for short filters, it is appropriate to apply a specific class of fast algorithms to the computation of the DWT for

Filter Length L	Direct Filter Bank $nOPS_{direct}$ (4.2)	FFT-based $nOPS_{FFT}$ (4.17)	Short-length $nOPS_{short}$ (4.19)	
			Decomposition	
4	$(4 \oplus 3) = 7$	$(8 \oplus 10) = 18$	(2×2)	$(3 \oplus 4) = 7$
6	$(6 \oplus 5) = 11$	$(7.84 \oplus 12.17) = 20.01$	(3×2)	$(4 \oplus 6.3) = 10.3$
8	$(8 \oplus 7) = 15$	$(8 \oplus 14) = 22$	$(2 \times 2 \times 2)$	$(4.5 \oplus 8.5) = 13$
10	$(10 \oplus 9) = 19$	$(8.24 \oplus 15.53) = 23.78$	(5×2)	$(4.8 \oplus 10.2) = 15$
12	$(12 \oplus 11) = 23$	$(8.5 \oplus 16.84) = 25.35$	$(2 \times 3 \times 2)$	$(6 \oplus 12) = 18$
14	$(14 \oplus 13) = 27$	$(8.76 \oplus 17.99) = 26.74$		
16	$(16 \oplus 15) = 31$	$(9 \oplus 19) = 28$	$(2 \times 2 \times 2 \times 2)$	$(9 \oplus 13) = 21$
18	$(18 \oplus 17) = 35$	$(9.23 \oplus 19.91) = 29.14$	$(2 \times 3 \times 3)$	$(8 \oplus 17) = 25$
20	$(20 \oplus 19) = 39$	$(9.44 \oplus 20.73) = 30.18$	$(5 \times 2 \times 2)$	$(7.2 \oplus 21.4) = 28.6$
22	$(22 \oplus 21) = 43$	$(9.65 \oplus 21.48) = 31.13$		
24	$(24 \oplus 23) = 47$	$(9.84 \oplus 22.18) = 32.01$	$(2 \times 2 \times 3 \times 2)$	$(12 \oplus 18) = 30$
26	$(26 \oplus 25) = 51$	$(10.02 \oplus 22.82) = 32.83$		
28	$(28 \oplus 27) = 55$	$(10.19 \oplus 23.42) = 33.6$		
30	$(30 \oplus 29) = 59$	$(10.35 \oplus 23.97) = 34.32$	$(5 \times 3 \times 2)$	$(9.6 \oplus 27) = 36.6$
32	$(32 \oplus 31) = 63$	$(24.5 \oplus 24.5) = 35$	$(2 \times 2 \times 2 \times 2 \times 2)$	$(18 \oplus 22) = 40$

Table 4-1. Computational costs per point and per elementary cell for the direct filter bank algorithm (4.2), the FFT-based algorithm (4.17) and the fast running short-length FIR filtering algorithm (4.19).

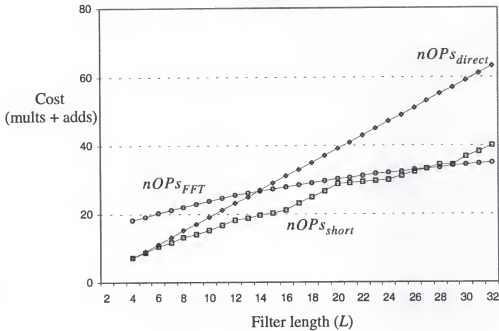


Figure 4-4. Comparison of computational cost of the direct filter bank method, the FFT version of the DWT algorithm, and the fast running short-length FIR filtering.

short filters. Although the resulting gain is fairly modest, it is still interesting when heavy computation of short filters is required [23]. Fast running short-length FIR algorithms [21] are the ones among the class. A general description of the algorithm follows: medium or long filters are decimated into short-length FIR filters with reduced computational cost where all multiplications are replaced by decimated subfilters. Since the process can be repeated on the subfilters, the short filters become basic elements in deriving these fast algorithms. One good aspect of the fast running FIR algorithms is that they allow various trade-offs between structural regularity and arithmetic efficiency. Figure 4-5 [23] shows a simple example of the short-length FIR filtering algorithm with the decimation ratio is two.

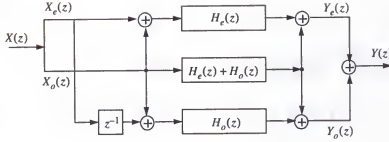


Figure 4-5. An example of fast running FIR filtering algorithm for short-filters. Here decimation ratio is two.

Let R be a decimation ratio and L the length of a filter where $L = kR$, and k is a positive integer. We denote by $F(R, L)$ an algorithm computing R outputs of a length- L FIR filter. We denote by $M_{R,L}^{short}$ and $A_{R,L}^{short}$ the number of multiplications and additions, respectively, of the $F(R, L)$ per output of a length- L filter. L can be factored by various R 's such that

$$L = R_1 R_2 \dots R_i \dots R_r L_r, \quad (4.18)$$

where r is a positive integer. Then $M_{R,L}^{short}$ and $A_{R,L}^{short}$ can be written as [21]

$$\begin{aligned} M_{R,L}^{short} &= M_r^{direct} \prod_{i=1}^r M_{R_i, R_i}^{short}, \\ A_{R,L}^{short} &= \sum_{i=1}^r A_{R_i}^{short} \sum_{j=1}^{i-1} M_{R_j, R_j}^{short} + (M_r^{direct} - 1) \prod_{i=1}^r M_{R_i, R_i}^{short}, \end{aligned} \quad (4.19)$$

where M_r^{direct} denotes the cost of the direct filter bank algorithm for length- r filters (4.1).

Mou and Duhamel [21] developed the short-length FIR filtering structures to implement algorithms of $F(2, 2)$, $F(3, 3)$ and $F(5, 5)$ that are most useful short-length filters

without any expense of structural regularity. Their computational costs are given by

$$\begin{aligned}
 (M_{2,2}^{short} \oplus A_{2,2}^{short}) &= (1.5 \oplus 2) , \\
 (M_{3,3}^{short} \oplus A_{3,3}^{short}) &= (2 \oplus 3.3) , \\
 (M_{5,5}^{short} \oplus A_{5,5}^{short}) &= (2.4 \oplus 8) .
 \end{aligned} \tag{4.20}$$

Table 4-1 lists the resulting costs of the short-length FIR filtering algorithm. The result shows that the short-length FIR algorithm is more efficient than the FFT-based DWT algorithms for lengths up to $L = 30$. Rioul and Duhamel [23] pointed out that the short-length FIR algorithms could save computational cost up to 30% for short-length filters. Since practical applications generally use short filters to compute the DWT's, such a gain can make a huge difference in the performance of the DWT's when heavy computation is involved.

CHAPTER 5

MULTISCALE BLOCK ALGORITHM

In this chapter we describe the design of a high-speed algorithm to support interactive analysis via multiscale processing. In addition to writing multi-threaded code, based on kernel support for multiple threads of control and changing the scheduling class policy to real-time mode, our algorithm uses hardware and software support for high-speed image manipulation, and achieves the best performance from the combination of all. This algorithm is utilized later to provide the best performance in computing the discrete wavelet transform for overcomplete representations.

We first review the architecture of the SPARCstation SX system which we have used as a base platform in developing most of our algorithms. And we describe briefly XIL foundation graphics interface and relevant functions accelerated on the SX system processor. We then introduce a new algorithm: *multiscale block algorithm*, to accomplish the forward and inverse discrete dyadic wavelet transforms that exploits the parallel architecture of the machine. We show how this multiscale block algorithm works and demonstrate the amount of speedup achieved over traditional linear convolution schemes and conventional FFT methods.

5.1 Architectural Overview of SPARCstation SX System

The SPARCstation SX graphics processor is a scalable graphics architecture consisting of an enhancement to memory controller chip on an MBus within a

SPARCstation. The SPARCstation SX system relies on the notion that graphics operations become extensions of native integer and floating-point operations of the CPU. The SPARCstation SX system employs a closely-coupled architecture, wherein a dedicated processor handles low-level display operations through processor-memory direct access. Residing directly on the CPU-memory bus, the SPARCstation SX memory controller, called SMC, can perform extremely fast transfer of data from physical memory (DRAM) to the display frame buffer. The close-coupling of the SMC to the SPARCstation CPU further enables floating point-based computation to utilize the high-performance transformation capabilities on the SPARC processor. Figure 5-1 illustrates the high-level architecture of the SPARCstation MBus system, including CPU, SMC processor, and video memory.

The SMC processor lies on the MBus between the SPARC CPUs and memory. In addition, Video RAM (VRAM) is mapped into the same address space, allowing the SMC to access and manipulate the display in the same manner as DRAM. This addressing gives the SPARCstation SX system superior bandwidth in moving large size images, for example, mammograms, from memory to the screen. Instructions executing in one or more of the SPARC CPUs make requests of the memory controller (MC) portion of the processor. The memory controller logic fulfills CPU requests for memory accesses, such as those required for fetching, loading and storing instructions.

When one of the CPUs encounters a graphics operation, it passes it to the SMC pixel processor (SXPP), which executes the instruction directly. When memory access is required to complete an operation, the SXPP makes a request to the memory controller.

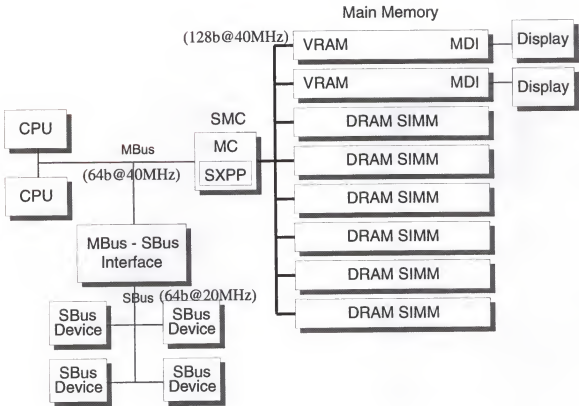


Figure 5-1. SPARCstation SX system high-level architecture [26].

5.2 XIL Imaging Library

The XIL library from Sun Microsystems is a foundation-level imaging interface providing common functions required by most imaging applications. Such a library is an application programming interface (API), but it is different from normal APIs because it is hardware dependent. The XIL library contains three primary components: a programming interface specification for basic imaging functionality, a high-performance implementation of the specification, and a standard hardware interface specification,

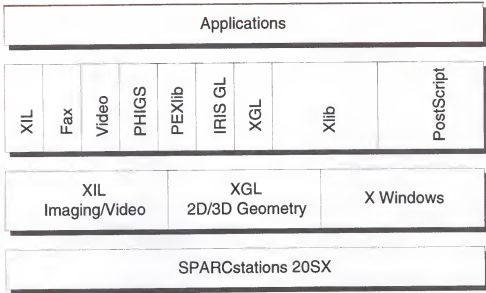


Figure 5-2. Solaris foundation graphics libraries and layered interface.

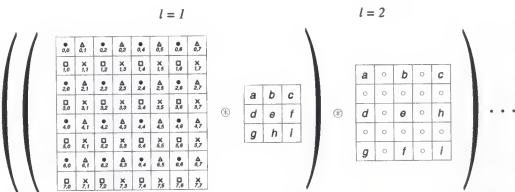
which enables third-party hardware developers to readily support the XIL library and applications built on it [28]. Figure 5-2 illustrates the XIL library in relationship to other foundation-level interfaces.

Monadic and *dyadic* operations are of particular interest within XIL library. Every XIL operator can be considered an atom. Groups of XIL operators concatenated in a sequential manner are known as molecules which perform the tasks of two or more atomic functions. Molecules are important to the SPARCstation SX SMC processor because they allow the device to notify the XIL library at runtime, and obtain hardware support for groups of chained operations. At runtime, the SMC device handler loads into memory and notifies the XIL library of the atoms and molecules it wishes to replace with its own equivalent mapping to hardware. The XIL library uses a directed acyclic graph to hold a

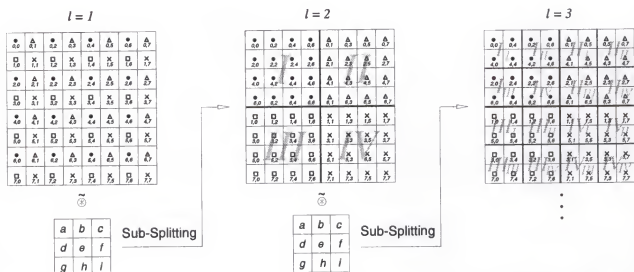
list of these dependencies during execution. When the XIL library encounters a known sequence of operations, it transfers the operation to the SMC for execution. XIL applications can realize significant performance improvement from the deferred execution.

5.3 Multiscale Block Algorithm

In frame-based multi-scale representations, spatial convolution require $2^l - 1$ zeros between each non-zero filter coefficients in a filter kernel as scale changes. This increases the computational cost exponentially. However, there exists an algorithm that reduces the cost by a factor of 2^l . The basic idea behind this algorithm is to keep the size of the filter kernels unchanged throughout multiscale analysis. Instead, we partition the input image into four sub-regions so that each region contains only those pixels affected by the filter kernel (without expansion). Figure 5-3 illustrates differences between the two approaches: a linear convolution with zero-paddings in the kernel, and an alternative method by splitting the convolved image into four sub-regions for further convolution at finer levels. The former is shown in Figure 5-3(a). At level 2, zeroes are inserted between the non-zero filter coefficients. Figure 5-3(b) graphically illustrates the new method. First, the input image is convolved with the original filter kernel at level 1. The convolved image is then divided into four sub-regions such that each region consists of only those pixels processed by the convolution at the next level ($l = 2$). After this splitting, convolutions are then applied to each of the sub-regions independently. To visually describe this, we show within the input image matrix four distinct pixel shapes: dot, triangle, square, and cross, in such a way that pixels that belong to the same group are identified.



(a) Traditional linear convolution at scale $l = 1$, $l = 2$. One zero is padded between non-zero filter coefficients in the filter kernel at scale 2.



(b) Alternative approach that avoids zero-paddings between the non-zero filter coefficients. We first convolve the input image with the original filter kernel at the level 1. The convolved image is then divided into four sub-regions, marked in the background as *I*, *II*, *III*, and *IV* with half-grayed tone, each of which holds pixels that are precisely affected by the filter kernel coefficients at the next level $l = 2$. Four distinct pixel shapes inside the input image are shown: *bullet*, *triangle*, *Box* and *cross*. Only pixels that have the same shape are processed by convolution at the next level.

Figure 5-3. Difference between a linear convolution with zero-paddings in the kernel (a) and an alternative method with four sub-regions (b).

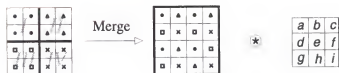


Figure 5-4. Multiresolution block convolution for image synthesis. Four sub-regions in the input image at a coarser level, marked as $\{I, II, III\}$, and $\{IV\}$, are merged into a larger region at a finer level. Then, convolution is applied to the merged region.

For example, let us compute the value located at coordinate $(2, 2)$ in the input image shown in Figure 5-3(a) marked by a dot. All pixels within the rectangular area from $(0, 0)$ to $(4, 4)$ will participate in the computation. However, with respect to traditional convolution the zeros in the filter kernel at level 2 effectively mask out all other pixels having shapes other than a dot. Thus pixels that actually take part in the computation are those having a dot shape located at coordinates $(0, 0)$, $(0, 2)$, $(0, 4)$, $(2, 0)$, $(2, 2)$, $(2, 4)$, $(4, 0)$, $(4, 2)$ and $(4, 4)$. The region labeled I shown in Figure 5-3(b) at the level 2 contains exactly those pixels. The convolution is carried out separately in this region and gives us the same result numerically as the traditional method.

Synthesis is simply the reverse of analysis. In other words, we first merge four sub-regions at a coarser level into a larger region, then convolve it with a filter kernel. Figure 5-4 shows the synthesis process.

For quantitative comparison, let us compute the computational cost of each method of implementation. For simplicity, we focus on the analysis only. Let the size of the input image and the size of the original filter kernel be n and m , respectively. And let us denote the level of analysis by l . Then, in the traditional linear convolution case, the

computational cost C_A is

$$C_A = n^2 \cdot (m + 2^l - 2)^2,$$

and in our approach, the computational cost C_B is

$$C_B = n^2 \cdot 2^{l-1} \cdot m^2.$$

Normally, the size of the filter kernel is small, so we can assume m is a constant. The speedup by the new approach is then

$$O\left(\frac{C_A}{C_B}\right) = O\left(\frac{2^{2l}}{2^l}\right) = O(2^l).$$

Algorithm 1 computes the two-dimensional discrete wavelet transform by this more efficient method. The variables, n and m , denote the size of the input image and the size of the filter kernel, respectively. $[Filter]_{X,Y}$ is created by the tensor product of two 1D filter kernels: X, Y . Thus,

$$[Filter]_{X,Y} = X'Y.$$

$[X]_{r,c}^{w,h}$ defines a ROI of size $w \times h$ starting at $[r \times w, c \times h]$. There are three global variables in the algorithm: *LEVEL*, *WIDTH*, *HEIGHT*. By *LEVEL*, we denote the number of levels for the transform, and by *WIDTH* and *HEIGHT* the width and height of the input image, respectively.

Algorithm 2 describes our algorithm for computing two-dimensional inverse discrete dyadic wavelet transform using the method shown in Figure 5-4. $\overline{[Filter]}$ denote the complex conjugate of $[Filter]$.

We show visually in Figure 5-5 the process of a two dimensional forward wavelet transform using the method described in **Algorithm 1**. In Figure 5-5(b), we display a

Algorithm 1 2-D discrete dyadic wavelet transform using block splitting method

Require n : dyadic number ($n > m$)

for $l = 0$ **to** $l < LEVEL$ **do**

$$w = \frac{WIDTH}{2^l}, h = \frac{HEIGHT}{2^l}$$

for $r = 0$ **to** $r < 2^l$ **do**

for $c = 0$ **to** $c < 2^l$ **do**

$[ROI]_{r,c}^{w,h} \Leftarrow$ Obtain a region of interest from the input image with
size of $w \times h$ starting at $[r \times w, c \times h]$

$[W_{2^l}^1]_{r,c}^{w,h} \Leftarrow \text{Split} ([ROI]_{r,c}^{w,h} * [Filter]_{L,G})$

$[W_{2^l}^2]_{r,c}^{w,h} \Leftarrow \text{Split} ([ROI]_{r,c}^{w,h} * [Filter]_{G,L})$

$[ROI]_{r,c}^{w,h} \Leftarrow \text{Split} ([ROI]_{r,c}^{w,h} * [Filter]_{H,H})$

end for

end for

end for

Algorithm 2 2-D discrete inverse dyadic wavelet transform by block merging sub-regions

Require n : dyadic number ($n > m$)

for $l = LEVEL - 1$ **downto** $l = 0$ **do**

$$w = \frac{WIDTH}{2^l}, h = \frac{HEIGHT}{2^l}$$

for $r = 0$ **to** $r < 2^l$ **do**

for $c = 0$ **to** $c < 2^l$ **do**

$[ROI]_{r,c}^{w,h} \Leftarrow \text{Merge sub-regions of } ([ROI]_{r,c}^{w,h} * \overline{[Filter]_{H,H}})$

$[W_{2^l}^1]_{r,c}^{w,h} \Leftarrow \text{Merge sub-regions of } ([W_{2^l}^1]_{r,c}^{w,h} * \overline{[Filter]_{L,G}})$

$[W_{2^l}^2]_{r,c}^{w,h} \Leftarrow \text{Merge sub-regions of } ([W_{2^l}^2]_{r,c}^{w,h} * \overline{[Filter]_{G,L}})$

$[ROI]_{r,c}^{w,h} \Leftarrow [ROI]_{r,c}^{w,h} + [W_{2^l}^1]_{r,c}^{w,h} + [W_{2^l}^2]_{r,c}^{w,h}$

end for

end for

end for

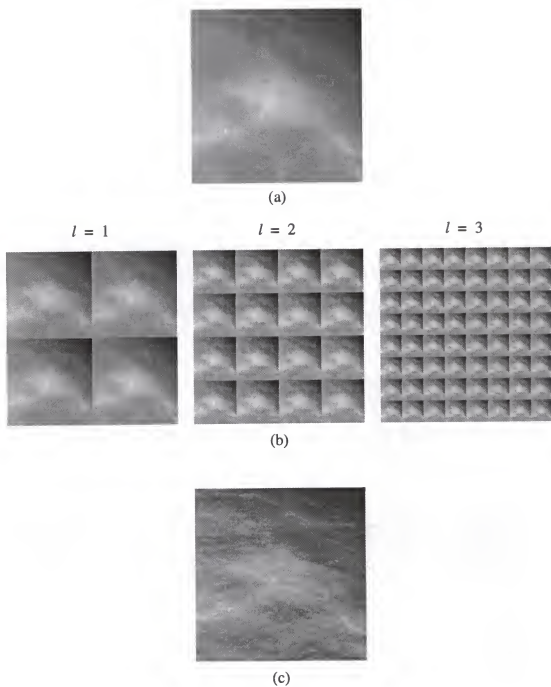


Figure 5-5. (a) Selected ROI containing an ill-defined mass. (b) DC components at levels $l = 1$, $l = 2$ and $l = 3$ after applying Algorithm 1. (c) Processed ROI with suspicious area enhanced.

<i>level</i>	<i>Multiscale Block</i>	<i>Traditional convolution</i>	<i>FFT approach</i>
1	1.18	0.83	13.75
2	1.81	2.87	27.46
3	3.71	7.87	41.05
4	6.29	26.01	54.65
5	13.03	93.93	68.62

Table 5-1. Computational costs in time (seconds) of splitting (in decomposition) and merging (in reconstruction) schemes versus traditional linear convolution with zero-padding between each non-zero filter coefficients and FFT implementations.

series of DC components at levels $l = 1$, $l = 2$ and $l = 3$. Finally, Figure 5-5(c) shows an enhanced version of the selected ROI shown in Figure 5-5(a).

We show in Table 5-1 the computational costs in time (second) for enhancement between methods of convolution by zero-padding between non-zero filter coefficients, forward and inverse FFT, and our new multiscale approach of splitting and merging sub-regions. Our sample input ROI was 256×256 and was computed up to 5 levels of analysis. The size of the original kernels used for the decomposition was 3×3 for both horizontal and vertical directional filters. In the reconstruction case, the kernels were 3×5 in the horizontal direction and 5×3 in the vertical direction. Figure 5-6 shows graphically comparison of the costs between the multiscale block algorithm, traditional convolution and FFT-base approach.

The costs shown in the Table 5-1 were measured on a SPARCstation 20 Model 71 from Sun Microsystems having a 75MHz SuperSPARC-II processor with 1MB

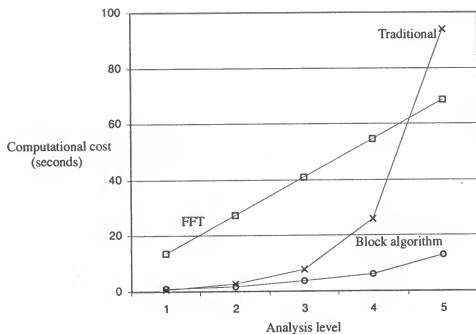


Figure 5-6. Comparison of computational costs between multiscale block, traditional convolution and FFT-base approach.

SuperCache, 96MB RAM, and 4MB (60ns) SX frame buffer. This table clearly shows that our approach is more efficient than existing convolution schemes and exceeded an FFT by a factor of 5 for five levels of processing.

CHAPTER 6

LIFTING SCHEME

In this chapter we review the basic idea behind the lifting scheme introduced by Wim Sweldens [29] [30] for constructing compactly supported biorthogonal wavelets. We first study in Section 6.1 the properties of second generation wavelets which emerged from the need of more general basis functions in applications on general domains. In Section 6.2, we review the biorthogonal wavelets and duals to see how they are related each other to satisfy a perfect reconstruction condition in a filter bank coding system. In Section 6.3 we describe the mathematical background of lifting scheme and its definition. We then discuss in Section 6.4 the relation between interpolating scaling functions, associated wavelets and the lifting scheme. Finally in Section 6.5 we show how this lifting scheme can be generalized to construct any wavelet or wavelet transform with a finite number of lifting steps [11].

6.1 Second Generation Wavelets

Wavelets $\psi_{j,k}$ are traditionally defined by translates and dilates of one particular function, a *mother* wavelet ψ , such that $\psi_{j,k}(x) = \psi(a^j x - k)$. These wavelets are referred to as first generation wavelets. The properties of first generation wavelets are the following: They form a Riesz basis for $L_2(R)$, so that a function f in $L_2(R)$ can be represented with the wavelet basis $\{\psi_{j,k}\}$ as

$$f = \sum_{j,k} \gamma_{j,k} \Psi_{j,k}$$

where the coefficients $\gamma_{j,k}$ are computed by $\gamma_{j,k} = \langle f, \tilde{\Psi}_{j,k} \rangle$. $\tilde{\Psi}$ is a dual wavelet of Ψ . The wavelets and their duals are well localized in space and frequency. In other words, the wavelets and duals have compact supports, which are smooth (i.e., they decay towards high frequencies), and have vanishing polynomial moments, which decay towards low frequency. As discussed in the previous chapter, wavelets are also utilized into a framework of multiresolution analysis. Finally, they mostly rely on the Fourier transform as a basic instrumental tool since the operations of the wavelet family defined by translates and dilates become algebraic operations in Fourier space.

On top of the properties of the traditional wavelets, second generation wavelets have emerged from the need of more generalized analyzing functions which are not necessarily translates and dilates of one basis function. Some applications on general domains require wavelets that are defined on arbitrary, possibly non-smooth, domains of \mathbb{R}^n . A special case is the construction of wavelets on bounded domains without introducing end effects near boundaries. Some special wavelets are also required in analyzing data on curves, surfaces or manifolds independent of parametrization. There are also problems in practical applications where analysis of irregularly sampled data may be necessary, which can not be achieved by first generation wavelets and traditional wavelet transform algorithms.

Because of the non-translation invariance property of its basis functions, second generation wavelets can no longer use the Fourier transform as its construction tool. A question then arises: How may the second generation wavelets be constructed? The lifting scheme introduced by Wim Sweldens [29] [30] provides a profound answer. The scheme

was initially introduced as a tool in constructing biorthogonal wavelets. Daubechies and Sweldens [11] later generalized the lifting scheme for the construction of any wavelet by factoring wavelet transforms through a finite number of lifting steps. In the next section we first study the properties of the biorthogonal wavelets and duals which is the first footstone in developing the lifting scheme.

6.2 Biorthogonal Wavelets and Duals

The orthogonality property in wavelets imposes a strong limitation in the construction of wavelets. The Haar wavelet is the only real-valued wavelet which is compactly supported, orthogonal and symmetric at the same time [9]. The scaling function $\varphi \in L_2$, and the wavelet function $\psi \in L_2$ both contribute to a multiscale analysis and must satisfy the refinement relations in the sense that,

$$\begin{aligned}\varphi(x) &= 2 \sum_k h_k \varphi(2x - k), \\ \psi(x) &= 2 \sum_k g_k \varphi(2x - k).\end{aligned}\tag{6.1}$$

The set of translated scale functions $\{\varphi(x - k) | k \in \mathbb{Z}\}$ form a Riesz basis. The refinement relations are also applicable to their duals given as $\tilde{\varphi}$ and $\tilde{\psi}$, with coefficients \tilde{h}_k and \tilde{g}_k respectively, such that

$$\begin{aligned}\tilde{\varphi}(x) &= 2 \sum_k \tilde{h}_k \tilde{\varphi}(2x - k), \\ \tilde{\psi}(x) &= 2 \sum_k \tilde{g}_k \tilde{\varphi}(2x - k).\end{aligned}$$

These duals also fit into a framework of multiresolution analysis. The duals are biorthogonal to their primals in the sense that,

$$\begin{aligned}
\langle \tilde{\phi}(x), \psi(x-k) \rangle &= \langle \tilde{\psi}(x), \phi(x-k) \rangle = 0, \\
\langle \tilde{\phi}(x), \phi(x-k) \rangle &= \langle \tilde{\psi}(x), \psi(x-k) \rangle = \delta_k.
\end{aligned} \tag{6.2}$$

The biorthogonality condition (6.2) is equivalent to

$$\begin{aligned}
\sum_k \hat{\phi}(\omega + 2k\pi) \overline{\hat{\phi}(\omega + 2k\pi)} &= \sum_k \hat{\psi}(\omega + 2k\pi) \overline{\hat{\psi}(\omega + 2k\pi)} = 1, \\
\sum_k \hat{\psi}(\omega + 2k\pi) \overline{\hat{\phi}(\omega + 2k\pi)} &= \sum_k \hat{\phi}(\omega + 2k\pi) \overline{\hat{\psi}(\omega + 2k\pi)} = 0.
\end{aligned} \tag{6.3}$$

Combining the conditions (6.2), (6.3) and the refinement relation (6.1), we can derive a biorthogonal condition for FIR filters, h , g , \tilde{h} , and \tilde{g} , as

$$\begin{aligned}
\tilde{h}(\omega) \overline{h(\omega)} + \tilde{h}(\omega + \pi) \overline{h(\omega + \pi)} &= 1 \\
\tilde{g}(\omega) \overline{g(\omega)} + \tilde{g}(\omega + \pi) \overline{g(\omega + \pi)} &= 1 \\
\tilde{g}(\omega) \overline{h(\omega)} + \tilde{g}(\omega + \pi) \overline{h(\omega + \pi)} &= 0 \\
\tilde{h}(\omega) \overline{g(\omega)} + \tilde{h}(\omega + \pi) \overline{g(\omega + \pi)} &= 0,
\end{aligned} \tag{6.4}$$

or equivalently in matrix notation,

$$\begin{bmatrix} \tilde{h}(\omega) & \tilde{h}(\omega + \pi) \\ \tilde{g}(\omega) & \tilde{g}(\omega + \pi) \end{bmatrix} \begin{bmatrix} \overline{h(\omega)} & \overline{g(\omega)} \\ \overline{h(\omega + \pi)} & \overline{g(\omega + \pi)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{6.5}$$

Hence, if we denote modulation matrices, $m(\omega)$ and $\tilde{m}(\omega)$, respectively by

$$m(\omega) = \begin{bmatrix} h(\omega) & h(\omega + \pi) \\ g(\omega) & g(\omega + \pi) \end{bmatrix} \quad \text{and} \quad \tilde{m}(\omega) = \begin{bmatrix} \tilde{h}(\omega) & \tilde{h}(\omega + \pi) \\ \tilde{g}(\omega) & \tilde{g}(\omega + \pi) \end{bmatrix},$$

then, the biorthogonality condition (6.5) becomes

$$\forall \omega \in \mathbf{R}, \quad \tilde{m}(\omega) \overline{m^t(\omega)} = \mathbf{I}. \tag{6.6}$$

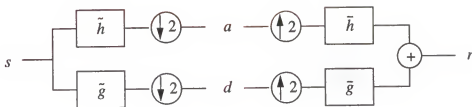


Figure 6-1. A two channel subband coding scheme.

By transposing equation (6.5), we see that

$$\begin{aligned} \overline{h(\omega)}\tilde{h}(\omega) + \overline{g(\omega)}\tilde{g}(\omega) &= 1 \\ \overline{h(\omega)}\tilde{h}(\omega + \pi) + \overline{g(\omega)}\tilde{g}(\omega + \pi) &= 0. \end{aligned} \quad (6.7)$$

Equation (6.7) can also be driven by a general two channel subband coding scheme, shown in Figure 6-1. A discrete input signal $s(n)$ is decomposed into an approximation signal $a(n)$ and a detail signal $d(n)$. They are defined by

$$\begin{aligned} a(2\omega) &= \frac{1}{2}(\tilde{h}(\omega)s(\omega) + \tilde{h}(\omega + \pi)s(\omega + \pi)), \\ d(2\omega) &= \frac{1}{2}(\tilde{g}(\omega)s(\omega) + \tilde{g}(\omega + \pi)s(\omega + \pi)). \end{aligned} \quad (6.8)$$

The reconstructed signal in the frequency domain can be written by

$$r(\omega) = \alpha(\omega)s(\omega) + \beta(\omega)s(\omega + \pi), \quad (6.9)$$

where

$$\begin{aligned} \alpha(\omega) &= \overline{h(\omega)}\tilde{h}(\omega) + \overline{g(\omega)}\tilde{g}(\omega), \\ \beta(\omega) &= \overline{h(\omega)}\tilde{h}(\omega + \pi) + \overline{g(\omega)}\tilde{g}(\omega + \pi). \end{aligned} \quad (6.10)$$

In order to achieve perfect reconstruction in subband coding system, we keep only the linear shift-invariant (LSI) system response appearing in the first term of equation (6.9)

and cancel out the system aliasing in the second term of equation (6.9). From $\alpha(\omega) = 1$ and $\beta(\omega) = 0$, we can derive (6.7).

By applying Cramer's rule to equation (6.7), we observe that

$$\tilde{h}(\omega) = \frac{\overline{g(\omega + \pi)}}{D_m(\omega)}, \quad \text{and} \quad \tilde{g}(\omega) = -\frac{\overline{h(\omega + \pi)}}{D_m(\omega)}, \quad (6.11)$$

where $D_m(\omega) = \det m(\omega)$. In the case of finite filters, we need to restrict that $D_m(\omega)$ to be a monomial, and we choose

$$D_m(\omega) = -e^{-i\omega}. \quad (6.12)$$

By applying equation (6.12) to equations (6.11) and (6.7), we have

$$\tilde{g}(\omega) = -e^{-i\omega} \overline{h(\omega + \pi)} \quad \text{and} \quad g(\omega) = -e^{-i\omega} \overline{\tilde{h}(\omega + \pi)}, \quad (6.13)$$

and equation (6.7) becomes

$$\overline{h(\omega)} \tilde{h}(\omega) + \overline{h(\omega + \pi)} \tilde{h}(\omega + \pi) = 1. \quad (6.14)$$

The filters satisfying (6.14) are called conjugate filters.

After finding the conjugate filters, we can construct a scaling function. By the refinement relation (6.1), the Fourier transform of the scaling function is then defined by

$$\hat{\phi}(\omega) = H\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right), \quad (6.15)$$

where H is the Fourier transform of the finite filter h , in other words

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n) e^{-in\omega}.$$

Since $\hat{\phi}(0) = 1$, infinite iterations of equation (6.15) with a conjugate filter H satisfying

condition (6.14) yields the limiting scaling function

$$\hat{\phi}(\omega) = \prod_{j=1}^{\infty} H(2^{-j}\omega). \quad (6.16)$$

A similar statement holds for the dual scaling function $\tilde{\phi}$ and its associated dual filter \tilde{H} . Since the properties of the filter H determine the properties of the scaling function such as the smoothness and its asymptotic decay at infinity, we do not need the scaling function itself in many applications, rather we directly work with the finite filter H .

Figure 6-2 shows the limiting process to make the Daubechies's four tap scaling

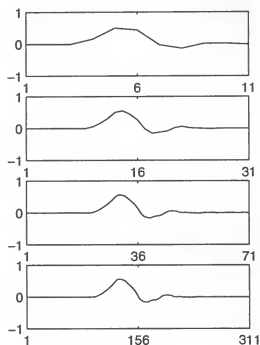


Figure 6-2. Graph showing the limiting process to make the Daubechies tap 4 scaling function. From top to bottom, scaling functions without iteration, after one iteration, after two iterations, and after fifteen iterations, respectively.

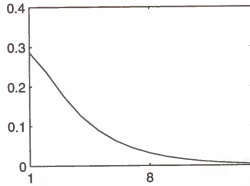


Figure 6-3. Graph of root mean square errors involved in the limiting steps to make the Daubechies's tap 4 scaling function shown in Figure 6-2.

function. From top to bottom in Figure 6-2, they are scaling functions without iteration, after one iteration, after two iterations and after fifteen iterations, respectively. Figure 6-3 displays a graph of root mean square errors involved in the limiting steps to make the Daubechies's four tap scaling function. The limiting process makes little difference after fifteen iterations ($\text{RMS} < 2.24e^{-16}$).

As pointed out by Cohen *et al.* [7], useful analyzing properties such as oscillations, and moments can be concentrated on the $\tilde{\psi}$ whereas the synthesis properties such as regularity are determined by ψ . $\tilde{\psi}$ and ψ can have very different regularity properties, however in practice, ψ is much more regular than $\tilde{\psi}$ [10]. In other words, analyzing dual wavelets have more vanishing moments than synthesizing wavelets.

6.3 Lifting Scheme

We recall the Vetterli-Herley lemma [32] which defines a complete characterization of filters, \tilde{h} and \tilde{h}^{old} , biorthogonal to a given filter, h . The two dual

filters \tilde{h} and \tilde{h}^{old} are biorthogonal to h in the sense of (6.14), and they are related by

$$\tilde{h}(\omega) = \tilde{h}^{old} + e^{-i\omega} \overline{\tilde{h}(\omega + \pi) s(2\omega)}, \quad (6.17)$$

where $s(\omega)$ is a trigonometric polynomial. In other words, if one of the dual filters is biorthogonal to h , and they are related through (6.17), the other dual is biorthogonal to h as well [29]. By combining the biorthogonal condition for basis functions (6.3) and (6.17), we can show the following relation for the associated FIR filters. From an initial set of finite biorthogonal filters $\{h, \tilde{h}^{old}, g^{old}, \tilde{g}\}$, a new set of finite biorthogonal filters can be found by

$$\begin{aligned} \tilde{h}(\omega) &= \tilde{h}^{old}(\omega) + \tilde{g}(\omega) \overline{s(2\omega)} \\ g(\omega) &= g^{old}(\omega) - h(\omega) s(2\omega). \end{aligned} \quad (6.18)$$

This is called *lifting* and this operation leads to the (primal) lifting scheme. The primal and dual basis functions with lifting are then given by:

$$\begin{aligned} \varphi(x) &= \varphi^{old}(x) \\ \tilde{\varphi}(x) &= 2 \sum_k \tilde{h}_k^{old} \tilde{\varphi}(2x - k) + \sum_k s_{-k} \tilde{\psi}(x - k) \\ \psi(x) &= \psi^{old}(x) - \sum_k s_k \varphi(x - k) \\ \tilde{\psi}(x) &= 2 \sum_k \tilde{g}_k^{old} \tilde{\varphi}(2x - k). \end{aligned} \quad (6.19)$$

There are several facts notable in (6.19). The scaling function does not change after lifting. The wavelet after lifting uses the scaling function in the same level. The wavelet construction formula tells us that the property of the wavelet after lifting is mostly determined by the property of the trigonometric polynomial s . The dual wavelet also changes. However since the changes in the dual wavelets are from the changes in the dual scaling functions, the changes of the dual wavelet are not meaningful by themselves. To

summarize, the lifting algorithm contributes to some fundamental and meaningful changes in the wavelet and the dual scaling functions through s which defines the true power of the lifting scheme. In other words, a multiresolution analysis can be achieved by starting from a simple basis function and building a more performant one with desirable properties.

We can also define a dual lifting from an initial set of filters $\{h^{old}, \tilde{h}, g, \tilde{g}^{old}\}$ and a dual lifting polynomial \tilde{s} such that

$$\begin{aligned} h(\omega) &= h^{old}(\omega) + g(\omega)\overline{\tilde{s}(2\omega)} \\ \tilde{g}(\omega) &= \tilde{g}^{old}(\omega) - \tilde{h}(\omega)\tilde{s}(2\omega) . \end{aligned} \quad (6.20)$$

The dual lifting scheme is thus defined by simply toggling the tildes in (6.19) as

$$\begin{aligned} \tilde{\varphi}(x) &= \tilde{\varphi}^{old}(x) \\ \varphi(x) &= 2 \sum_k \tilde{h}_k^{old} \varphi(2x-k) + \sum_k \tilde{s}_{-k} \psi(x-k) \\ \tilde{\psi}(x) &= \tilde{\psi}^{old}(x) - \sum_k \tilde{s}_k \tilde{\varphi}(x-k) \\ \psi(x) &= 2 \sum_k g_k^{old} \varphi(2x-k) . \end{aligned} \quad (6.21)$$

By alternating the primal and dual lifting scheme, one can come up with a multiresolution analysis geared for specific properties. This so-called *cakewalk* algorithm leads to a canonical form of the wavelet transform shown in Figure 6-4. Here the filters followed by successive subsampling define the *Lazy* wavelet transform which simply splits an original signal into odd and even subsets. The dual lifting predicts wavelet coefficients with the help of the scaling function coefficients based upon the correlation present in the finer signal. The primal lifting then updates the scaling function coefficients using the previously predicted wavelet coefficients to remove possible aliasing resulting from downsampling.

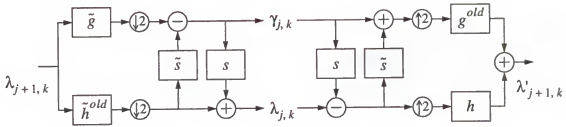


Figure 6-4. A 1-D discrete wavelet transform with lifting polynomials.

Based upon the formulae of the primal and dual lifting schemes, a fast forward and inverse wavelet transform can thus be formulated as

FORWARD TRANSFORM

$$(I) \quad \lambda_{j,l} = \sqrt{2} \sum_k \tilde{h}_{k-2l} \lambda_{j+1,k} \quad \text{and} \quad \gamma_{j,l} = \sqrt{2} \sum_k \tilde{g}_{k-2l} \lambda_{j+1,k},$$

$$(II) \quad \gamma_{j,l} = \gamma_{j,l} - \sum_k^{N-1} \tilde{s}_{l-k} \lambda_{j,k}$$

$$(III) \quad \lambda_{j,l} = \lambda_{j,l} + \sum_k^{\tilde{N}-1} s_{l-k} \gamma_{j,k}$$

INVERSE TRANSFORM

$$(I) \quad \lambda_{j,l} = \lambda_{j,l} - \sum_k^{\tilde{N}-1} s_{l-k} \gamma_{j,k}$$

$$(II) \quad \gamma_{j,l} = \gamma_{j,l} + \sum_k^{N-1} \tilde{s}_{l-k} \lambda_{j,k}$$

$$(III) \quad \lambda_{j+1,k} = \sqrt{2} \sum_l h_{k-2l} \lambda_{j,l} + \sqrt{2} \sum_l g_{k-2l}^{\text{old}} \gamma_{j,l}.$$

Here N and \tilde{N} are the orders of the primal and dual lifting polynomials defined as s and \tilde{s} respectively.

Now we can define a canonical implementation structure of the lifting scheme with three stages: *split*, *predict* and *update*, idea of which is well described in the *cakewalk* algorithm shown in Figure 6-4. A simpler version of the forward wavelet transform with lifting is shown in Figure 6-5. The split process splits the input data $\{\lambda_{j+1}\}$ into two smaller subsets $\{\lambda_j\}$ and $\{\gamma_j\}$. From the perfect reconstruction property in the polyphase representation, the most trivial subsets are even and odd samples. This is often referred to as the *Lazy* wavelet transform [29] as described earlier. In the second stage *Predict*, we try to use the $\{\lambda_j\}$ subset to predict the $\{\gamma_j\}$ subset based on the correlation present in the original data. The $\{\lambda_j\}$ subset is then *lifted* with the help of the wavelet coefficients $\{\gamma_j\}$. The idea is to find a better $\{\lambda_j\}$ so that a certain scalar quantity $Q()$, such as the mean is preserved, $Q(\lambda_{j+1}) = Q(\lambda_j)$. These aspects lead to the following forward wavelet transform:

$$\text{For } j = -1 \text{ downto } -n: \begin{cases} \{\lambda_j, \gamma_j\} = \text{Split}(\lambda_{j+1}) \\ \gamma_j = \gamma_j - P(\lambda_j) \\ \lambda_j = \lambda_j + U(\gamma_j) \end{cases}$$

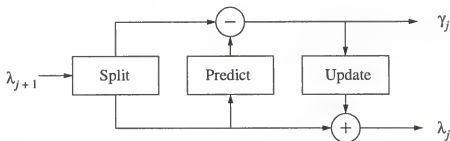


Figure 6-5. Canonical structure in implementing 1-D discrete forward wavelet transform with lifting.

where \mathcal{P} and \mathcal{U} are prediction and update operators, respectively. The inverse transform is immediately driven by reversing the operations and toggling the signs:

$$\text{For } j = -n \text{ upto } -1: \begin{cases} \lambda_j = \lambda_j - \mathcal{U}(\gamma_j) \\ \gamma_j = \gamma_j + \mathcal{P}(\lambda_j) \\ \lambda_{j+1} = \text{Join}(\lambda_j, \gamma_j) . \end{cases}$$

Figure 6-6 shows the wavelet representation of a ROI in a mammogram with lifting after level 3 decomposition. The wavelet transform presented here in fact is the $(N = 2, \tilde{N} = 2)$ biorthogonal wavelet transform of Cohen-Daubechies-Feauveau [7].

6.4 Interpolating Scaling Functions, Associated Wavelets and Lifting Scheme

Sweldens [29] [30] [31] showed that the lifting scheme described in the previous section is connected with interpolating scaling functions. The use of interpolating scaling functions in multiresolution analysis has been studied by several authors [12] [13]. The main advantage of the interpolating scaling function is that the coefficients λ_k in a function expansion $f(x) = \sum_k \lambda_k \varphi(x - k)$ can be easily obtained by $\lambda_k = f(k)$ since the interpolating scaling function is defined as $\varphi(k) = \delta_k$ for all $k \in \mathbb{Z}$. By combining the definition of the interpolating scaling function and the refinement relation (6.1), we get $h_{2k} = \frac{\delta_k}{2}$ which becomes

$$h(\omega) + h(\omega + \pi) = 1 . \quad (6.22)$$

Filters satisfying (6.22) is called interpolating filters. *à-trous* filters are the interpolating filters which are used to compute coefficients of a continuous wavelet transform [8] [24].

There have been several approaches to compute interpolating scaling functions. Here we briefly discuss one proposed by Deslauriers and Dubuc [12]: *interpolating subdivision*, which is connected to our *Overcomplete Lifting Scheme*. The interpolating

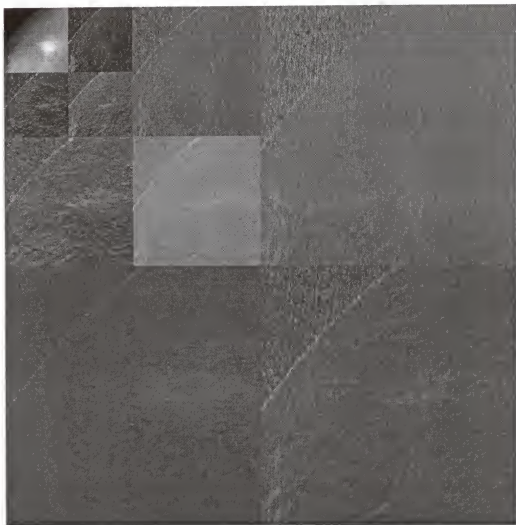


Figure 6-6. Wavelet representations of a ROI in a mammogram computed via lifting for three levels of decomposition.

subdivision is formally defined as follows. We first construct a polynomial P of degree $N-1$ so that $P(x_{j,k+l}) = \lambda_{j,k+l}$ for $-D+1 \leq l \leq D$, where $N = 2D$. We then calculate coefficients on the next finer level as the value of this polynomial at $\lambda_{j+1,2k+1} = P(x_{j+1,2k+1})$. Figure 6-7 shows the scaling functions with interpolating subdivision of order 2, 4, 6, and 8 from top to bottom.

These interpolating scaling functions have compact supports, and are symmetric, so that a $\phi(x)$ is zero outside the interval $[-N+1, N-1]$. The subdivision interpolating scaling functions and their translates reproduce polynomials up to degree $N-1$. They

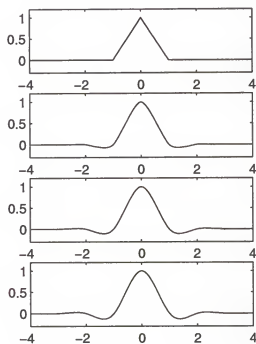


Figure 6-7. Interpolating scaling functions resulting from interpolating subdivisions of order 2, 4, 6 and 8 from top to bottom

also satisfy the refinement equation (6.1), which is critical in defining a multiresolution analysis. Since the order of the reproduced polynomials is invariant within an interval, the resulting scaling functions are well adapted on an interval of finite signal without extensions such as zero padding, periodization or reflection. Figure 6-8 shows an example of cubic ($N = 4$) scaling functions changing their shapes to accommodate themselves near the left boundary without degrading the regularity. The interpolation properties previously mentioned are still preserved well on the boundary for these scaling functions.

Table 6-1 lists the lifting coefficients needed for the interpolating subdivision

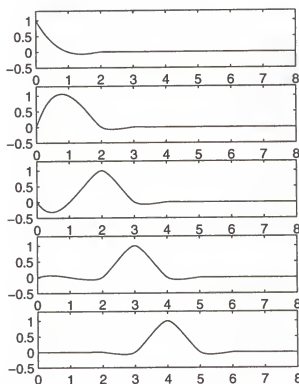


Figure 6-8. The cubic ($N = 4$) interpolating scaling functions near boundary (top to bottom) changing their shapes to alleviate boundary effects.

# of λ 's on (left, right)	Lifting Coefficients ($N = 2$)
(0, 2)	(-0.5, 1.5)
(1, 1)	(0.5, 0.5)
(2, 0)	(1.5, -0.5)

(a)

# of λ 's on (left, right)	Lifting Coefficients ($N = 4$)
(0, 4)	(2.1875, -2.1875, 1.3125, -0.3125)
(1, 3)	(0.3125, 0.9375, -0.3125, 0.0625)
(2, 2)	(-0.0625, 0.5625, 0.5625, -0.0625)
(3, 1)	(0.0625, -0.3125, 0.9375, 0.3125)
(4, 0)	(-0.3125, 1.3125, -2.1875, 2.1875)

(b)

# of λ 's on (left, right)	Lifting Coefficients ($N = 6$)
(0, 6)	(2.7070, -4.5117, 5.4141, -3.8672, 1.5039, -0.2461)
(1, 5)	(0.2461, 1.2305, -0.8203, 0.4922, -0.1758, 0.0273)
(2, 4)	(-0.0273, 0.4102, 0.8203, -0.2734, 0.0820, -0.0117)
(3, 3)	(0.0117, -0.0977, 0.5859, 0.5859, -0.0977, 0.0117)
(4, 2)	(-0.0117, 0.0820, -0.2734, 0.8203, 0.4102, -0.0273)
(5, 1)	(0.0273, -0.1758, 0.4922, -0.8203, 1.2305, 0.2461)
(6, 0)	(-0.2461, 1.5039, -3.8672, 5.4141, -4.5117, 2.7070)

(c)

# of λ 's on (left, right)	Lifting Coefficients ($N = 8$)
(0, 8)	(3.1421, -7.3315, 13.1968, -15.7104, 12.2192, -5.9985, 1.6919, -0.2095)
(1, 7)	(0.2095, 1.4663, -1.4663, 1.4663, -1.0474, 0.4888, -0.1333, 0.0161)
(2, 6)	(-0.0161, 0.3384, 1.0151, -0.5640, 0.3384, -0.1450, 0.0376, -0.0044)
(3, 5)	(0.0044, -0.0513, 0.4614, 0.7690, -0.2563, 0.0923, -0.0220, 0.0024)
(4, 4)	(-0.0024, 0.0239, -0.1196, 0.5981, 0.5981, -0.1196, 0.0239, -0.0024)
(5, 3)	(0.0024, -0.0220, 0.0923, -0.2563, 0.7690, 0.4614, -0.0513, 0.0044)
(6, 2)	(-0.0044, 0.0376, -0.1450, 0.3384, -0.5640, 1.0151, 0.3384, -0.0161)
(7, 1)	(0.0161, -0.1333, 0.4888, -1.0474, 1.4663, -1.4663, 1.4663, 0.2095)
(8, 0)	(-0.2095, 1.6919, -5.9985, 12.2192, -15.7104, 13.1968, -7.3315, 3.1421)

(d)

Table 6-1. Lifting coefficients based on the number of λ 's on its left and right. (a), (b), (c) and (d) show the filter coefficients for $N = 2$, $N = 4$, $N = 6$ and $N = 8$, respectively

scheme which are computed from *Neville's* algorithm. Table 6-1(a), (b), (c) and (d) show the filter coefficients for $N = 2$, $N = 4$, $N = 6$ and $N = 8$, respectively. Each lists all coefficients to compute values at every possible location newly introduced as a middle point at the next finer level.

By filling in the refinement relation in the interpolating case, we see that

$$\delta_{k,k'} = \varphi_{j,k}(x_{j,k'}) = \sum_l h_{j,k,l} \varphi_{j+1,l}(x_{j+1,2k'}) = h_{j,k,2k'},$$

which can be written as

$$\varphi_{j,k} = \varphi_{j+1,2k} + \sum_m h_{j,k,2m+1} \varphi_{j+1,2m+1}.$$

An approximation $A_j f$ from a finer one $A_{j+1} f$ is then built by simple subsampling, $\lambda_{j,k} = \lambda_{j+1,2k}$, followed by one step subdivision on A_j :

$$A_j f = \sum_k \lambda_{j,k} \varphi_{j,k} = \sum_k \lambda_{j+1,2k} \varphi_{j+1,2k} + \sum_k \sum_m \lambda_{j,k} h_{j,k,2m+1} \varphi_{j+1,2m+1}.$$

The detail, $A_{j+1} - A_j$, consists only of functions of the form $\varphi_{j+1,2m+1}$. Thus the wavelets are given by $\psi_{j,m} = \varphi_{j+1,2m+1}$. However, a necessary condition for the wavelets to form a Riesz basis is that (dual) vanishing moment \tilde{N} is at least one. A new wavelet with \tilde{N} is then formulated by following with lifting scheme:

$$\psi_{j,m} = \varphi_{j+1,2m+1} - \sum_{l=0}^{\tilde{N}-1} s_{j,m}^l \varphi_{j,m+l},$$

where the lifting coefficients $s_{j,m}^l$ are chosen so that

$$\int \psi_{j,m}(x) = 0, \int x \psi_{j,m}(x) = 0, \dots, \int x^{\tilde{N}-1} \psi_{j,m}(x) = 0.$$

In other words, the new wavelet is composed of the old wavelet, a subsampled scaling function located at odd indices, and its immediate neighboring scaling functions at

the same scale j . Figure 6-9 shows wavelets with two (dual) vanishing moments, $N = 2$, associated with interpolating scaling functions the order of the subdivision of which are 2, 4, 6, and 8 from top to bottom. Figure 6-10 shows the wavelets affected by the boundary. The cubic ($N = 4$) scaling functions are used to construct the boundary wavelets having $\tilde{N} = 2$ vanishing moments.

The following shows how to increase the number of vanishing moments of the Haar wavelet (simplest wavelet) from one to two with the lifting scheme. The initial filters associated with the Haar wavelet and the scaling function are given by:

$$\begin{aligned}\tilde{h}^{old}(\omega) &= h(\omega) = 1/2 + 1/2e^{-i\omega}, \\ \tilde{g}(\omega) &= g^{old}(\omega) = -1/2 + 1/2e^{-i\omega}.\end{aligned}$$

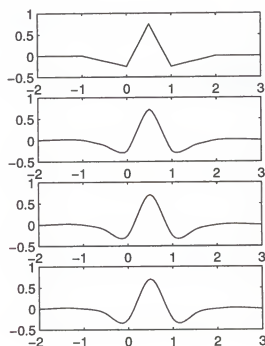


Figure 6-9. Wavelets with two (dual) vanishing moments ($\tilde{N} = 2$) associated with interpolating scaling functions whose orders of the subdivision are 2, 4, 6, and 8 from top to bottom.

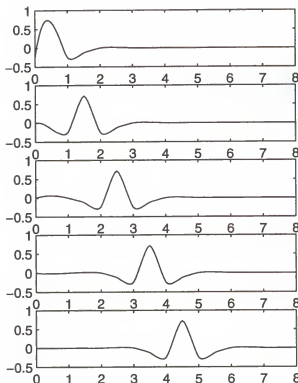


Figure 6-10. Wavelets accommodating the left boundary. Here $N = 4$ and $\tilde{N} = 2$.

After lifting, we have

$$g(\omega) = g^{old}(\omega) - h(\omega)s(2\omega).$$

In order to have two vanishing moments, $g(0) = 0$, and $g'(0) = 0$. By solving the linear equations, $s(\omega) = -i/4 \sin \omega$. This leads to one of the biorthogonal filters proposed by Cohen-Daubechies-Feauveau [7]:

$$\tilde{h} = -\frac{1}{16}e^{2i\omega} + \frac{1}{16}e^{i\omega} + \frac{1}{2} + \frac{1}{2}e^{-i\omega} + \frac{1}{16}e^{-2i\omega} - \frac{1}{16}e^{-3i\omega}. \quad (6.23)$$

Lifting (6.18) allows us to get a faster implementation of the wavelet transform if \tilde{h}^{old} is the shortest filter. The standard wavelet transform for the low-pass channel using

the filter (6.23) is

$$\lambda_{j,l} = -\frac{1}{16}\lambda_{j+1,2l-2} + \frac{1}{16}\lambda_{j+1,2l-1} + \frac{1}{2}\lambda_{j+1,2l} + \frac{1}{2}\lambda_{j+1,2l+1} + \frac{1}{16}\lambda_{j+1,2l+2} - \frac{1}{16}\lambda_{j+1,2l+3}. \quad (6.24)$$

Using the original filters associated with the Haar basis function and lifting coefficients with two vanishing moments, $s_1 = -s_{-1} = 1/8$, equation (6.24) becomes

$$\lambda_{j,l} = 1/2\lambda_{j+1,2l} + 1/2\lambda_{j+1,2l+1} + 1/8\gamma_{j,l-1} - 1/8\gamma_{j,l+1}, \quad (6.25)$$

which has fewer operations.

The complexity of the lifting scheme in this case is equivalent to the one required for the length-4 direct filter bank algorithm instead of the length-6's. Table 4-1 in Chapter 4 shows that $nOPs_{direct}|_{L=4} = (4 \oplus 3) = 7$, and $nOPs_{direct}|_{L=6} = (6 \oplus 5) = 11$. The speedup achieved by the lifting is thus the factor of 1.57. More details on the complexity of the lifting scheme will be studied later in this chapter.

6.5 Generalized Lifting Scheme

We have seen in the previous sections that there are many advantages of the lifting such as an in-place implementation of the fast wavelet transform, no need for the Fourier transform, easiness of the inverse transform, capability of custom-design of wavelets and adaptiveness of wavelet transforms. However the lifting scheme discussed was used in constructing biorthogonal wavelets. However can the lifting be used in constructing any wavelets or wavelet transforms with finite filters? Daubechies and Sweldens addressed the generality of the lifting by connecting the Euclidean algorithm to a subband transform in the z -domain [11].

The z -transform of a FIR filter h is a Laurent polynomial $H(z)$ given by

$$H(z) = \sum_{k=a}^b h_k z^{-k}$$

where a and b are respectively the lower and upper bound of the support of the filter h . The degree of the Laurent polynomial $H(z)$ is given by $|H(z)| = b - a$, so that a polynomial Cz^l has zero degree and is a monomial, where C is a non-zero constant and l is an integer. A Laurent polynomial is invertible if and only if it is a monomial.

The monomial constraint plays a key role in finding a modulation matrix or a polyphase matrix in a filter bank coding system. Figure 6-11 shows a discrete wavelet transform in z -domain. The conditions for perfect reconstruction are given by

$$\begin{aligned} H(z)\tilde{H}(z^{-1}) + G(z)\tilde{G}(z^{-1}) &= 2, \\ H(z)\tilde{H}(-z^{-1}) + G(z)\tilde{G}(-z^{-1}) &= 0. \end{aligned} \quad (6.26)$$

The modulation matrices $M(z)$ and $\tilde{M}(z)$ are defined as

$$M(z) = \begin{bmatrix} H(z) & H(-z) \\ G(z) & G(-z) \end{bmatrix}, \quad \tilde{M}(z) = \begin{bmatrix} \tilde{H}(z) & \tilde{H}(-z) \\ \tilde{G}(z) & \tilde{G}(-z) \end{bmatrix}. \quad (6.27)$$

By using these modulation matrices, the perfect reconstruction condition (6.26) can be

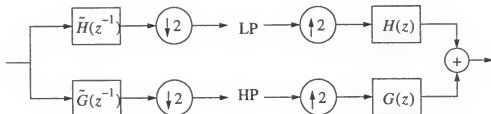


Figure 6-11. Discrete wavelet transform in z -domain.

written as

$$\tilde{M}(-z^{-1})^t M(z) = 2\mathbf{I}, \quad (6.28)$$

where \mathbf{I} is the 2 by 2 identity matrix.

We can simplify the perfect reconstruction in the filter bank structure using a polyphase representation which provides an efficient and convenient tool. The polyphase implementation works on different phases separately. The input is separated into even and odd phases followed by subsampling. Then the separate phases of the filters act in parallel on the separate phases of the input. Figure 6-12 shows a polyphase representation of the wavelet transform shown in Figure 6-11. The polyphase representation of a filter H is given as $H(z) = H_e(z^2) + z^{-1}H_o(z^2)$, where H_e and H_o contain respectively the even and odd coefficients. If we denote the polyphase matrix $P(z)$ as

$$P(z) = \begin{bmatrix} H_e(z) & G_e(z) \\ H_o(z) & G_o(z) \end{bmatrix}, \quad (6.29)$$

then the relation between the modulation matrix (6.27) and the polyphase matrix (6.29)

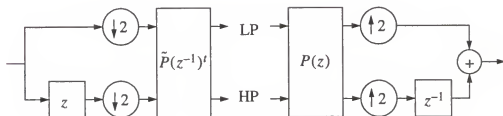


Figure 6-12. Polyphase representation of wavelet transform

becomes

$$P(z^2) = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ z & -z \end{bmatrix} M^t(z) = \frac{1}{2} \left(M(z) \begin{bmatrix} 1 & z \\ 1 & -z \end{bmatrix} \right)^t. \quad (6.30)$$

Now the perfect reconstruction property with modulation matrices (6.28) can be written in polyphase matrices as

$$P(z) \tilde{P}(z^{-1})^t = I. \quad (6.31)$$

From the invertible property of Laurent polynomials and (6.31), the determinant of $P(z)$ must be a monomial in z so that $P(z) = Cz^l$. In other words, finding FIR filters for a wavelet transform becomes finding a polyphase matrix $P(z)$ with the determinant one. By combining this monomial property and (6.31), the dual polyphase matrix $\tilde{P}(z)$ is given as

$$\tilde{P}(z) = \begin{bmatrix} G_o(z^{-1}) & -H_o(z^{-1}) \\ -G_e(z^{-1}) & H_e(z^{-1}) \end{bmatrix}.$$

The most trivial case of the polyphase matrix satisfying the condition of perfect reconstruction is $P(z) = I$ which implies $H(z) = \tilde{H}(z) = 1$ and $G(z) = \tilde{G}(z) = z^{-1}$. The transform with $P(z) = I$ is often referred as the *Lazy* wavelet transform since it does nothing but separating the input into even and odd subsets.

Using the polyphase structure, the lifting given in equation (6.18) can be written in z -domain as

$$\begin{aligned} \tilde{H}(z) &= \tilde{H}^{old}(z) + \overline{\tilde{G}(z)} S(z^{-2}) \\ G(z) &= G^{old}(z) - H(z) S(z^2), \end{aligned} \quad (6.32)$$

and the dual lifting in equation (6.20) as

$$\begin{aligned} H(z) &= H^{old}(z) + \overline{\tilde{G}(z)} \tilde{S}(z^{-2}) \\ \tilde{G}(z) &= \tilde{G}^{old}(z) - \tilde{H}(z) \tilde{S}(z^2). \end{aligned} \quad (6.33)$$

There are two *noble identities* for a decimator and interpolator. In most applications a decimator is preceded by a bandlimiting filter $H(z)$ to avoid aliasing. An interpolation filter, on the other hand, is followed by an interpolator to eliminate imaging effect. Figure 6-13 shows equivalent identity systems. In Figure 6-13(a), we have a decimator followed by a transfer function $F(z)$. This cascade is equivalent to the one in the right provided $F(z)$ is a rational transfer function, i.e. a ratio of polynomials in z^{-1} . In Figure 6-13(b), we show that the two cascades in the left and right are equivalent for a interpolator. Based on the property of the noble identities with $M = 2$ and $L = 2$, the polyphase components of $H(z)S(z^2)$ are $H_e(z)S(z)$ for even and $H_o(z)S(z)$ for odd. The new polyphase matrix after primal lifting is thus given by

$$P(z) = P^{old}(z) \begin{bmatrix} 1 & -S(z) \\ 0 & 1 \end{bmatrix}, \quad (6.34)$$

and the dual polyphase matrix by

$$\tilde{P}(z) = \tilde{P}^{old}(z) \begin{bmatrix} 1 & 0 \\ S(z^{-1}) & 1 \end{bmatrix}. \quad (6.35)$$

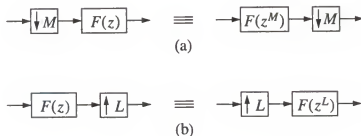


Figure 6-13. Noble identities for multirate systems. Identity systems for (a) a decimator, and (b) a interpolator.

Similarly, the new polyphase matrix after dual lifting is given as

$$P(z) = P^{old}(z) \begin{bmatrix} 1 & 0 \\ \tilde{S}(z) & 1 \end{bmatrix}, \quad (6.36)$$

and the dual polyphase matrix as

$$\tilde{P}(z) = \tilde{P}^{old}(z) \begin{bmatrix} 1 & -\tilde{S}(z^{-1}) \\ 0 & 1 \end{bmatrix}. \quad (6.37)$$

The Euclidean algorithm developed to find the greatest common divisor (GCD) of two natural numbers can be extended to find a GCD of two Laurent polynomials [11]. In other words, two Laurent polynomials $A(z)$ and $B(z)$ are presented as

$$\begin{bmatrix} A(z) \\ B(z) \end{bmatrix} = \prod_{i=1}^n \begin{bmatrix} Q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} A_n(z) \\ 0 \end{bmatrix} \quad (6.38)$$

where $A_n(z)$ is the GCD, n is the smallest number for which $B_n(z) = 0$, and $Q_i(z) = A_{i-1}(z)/B_{i-1}(z)$. Here if $A_n(z)$ is a monomial, then $A(z)$ and $B(z)$ are relatively prime. We can always choose the quotients $Q_i(z)$ so that the GCD $A_n(z)$ is a constant.

By applying the Euclidean algorithm to polyphase matrices, we can factor any pair of complementary filters $(H(z), G(z))$ into lifting steps. A filter pair $(H(z), G(z))$ is defined to be *complementary* if the determinant of the corresponding polyphase matrix $P(z)$ is one. The polyphase matrix can be given as

$$P(z) = \left(\prod_{i=1}^m \begin{bmatrix} 1 & -S_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \tilde{S}_i(z) & 1 \end{bmatrix} \right) \begin{bmatrix} K & 0 \\ 0 & \frac{1}{K} \end{bmatrix}, \quad (6.39)$$

and the dual polyphase matrix as

$$\tilde{P}(z) = \left(\prod_{i=1}^m \begin{bmatrix} 1 & 0 \\ S_i(z^{-1}) & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tilde{S}_i(z^{-1}) \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} \frac{1}{K} & 0 \\ 0 & K \end{bmatrix}. \quad (6.40)$$

Here K is a non-zero constant, and $m = \frac{n}{2} + 1$. The polyphase matrix for analysis filter bank $\tilde{P}(z^{-1})^t$ is thus written as

$$\tilde{P}(z^{-1})^t = \begin{bmatrix} \frac{1}{K} & 0 \\ 0 & K \end{bmatrix} \left(\prod_{i=m}^1 \begin{bmatrix} 1 & 0 \\ -\tilde{S}_i(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & S_i(z) \\ 0 & 1 \end{bmatrix} \right). \quad (6.41)$$

Figure 6-14 shows the forward wavelet transform using the dual polyphase matrix with finite lifting steps given in (6.41). Figure 6-15 represents the finite steps of the inverse wavelet transform using the polyphase matrix representation in (6.39).

Every wavelet or wavelet transform can thus be obtained with a factorization of the polyphase matrix representation of the filter bank by means of Euclidean algorithm.

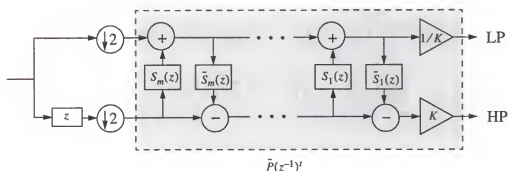


Figure 6-14. Forward wavelet transform using a dual polyphase matrix $\tilde{P}(z^{-1})^t$ with finite lifting steps.

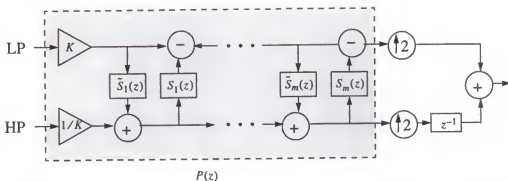


Figure 6-15. Inverse wavelet transform using the polyphase matrix $P(z)$ with finite lifting steps.

6.6 Computational cost of the Lifting Scheme

To compute the computational cost of the lifting scheme, we use the generic scheme presented in Figure 6-14. We need to find one or more lifting filters s and \tilde{s} , and normalization factors K and $1/K$, such that this scheme is equivalent to the one in Figure 6-1.

For the purpose of the cost comparison, we can rewrite the DWT algorithm using the lifting scheme with the following implementation steps. Since the inverse transform always follows immediately from the forward transform, we only focus on the forward transform.

1. Split

$$\lambda_i^{(0)} \leftarrow x_{2i}$$

$$\gamma_i^{(0)} \leftarrow x_{2i+1}$$

2. Lifting steps

for $l = 1$ to L

$$\text{Primal lifting: } \lambda_i^{(l)} \leftarrow \lambda_i^{(l-1)} + \sum_j \tilde{s}_j \gamma_j^{(l-1)}$$

$$\text{Dual lifting: } \gamma_i^{(l)} \leftarrow \gamma_i^{(l-1)} - \sum_j s_j \lambda_j^{(l-1)}$$

end for

3. Normalization

$$\lambda_i^{(L)} \leftarrow \frac{\lambda_i^{(L)}}{K}$$

$$\gamma_i^{(L)} \leftarrow K \gamma_i^{(L)}$$

Let L be the length of each of the two filters $h[n]$ and $g[n]$ used in the direct filter bank (standard) algorithm. As described in Chapter 4, the assumption of the same length of two filters does not lose generality since we can pad shorter filters with zeros to have the same length as the longest filter. Daubechies and Sweldens [11] showed that the computational cost of the lifting scheme with two length- L filters is $2L + 2$. Thus the cost of the lifting scheme per input point in the computational cell $nOPS_{lifting}$ is given as

$$nOPS_{lifting} = L + 1. \quad (6.42)$$

Since $nOPS_{direct} = 2L - 1$, the lifting scheme can achieve an overall speedup of 2 over the direct filter bank (standard) algorithm.

Figure 6-16 shows the comparison of the costs between the various DWT's described in Chapter 4, such as the direct filter bank algorithm, the FFT-based algorithm and the short-length FIR algorithm, and the complexity of the lifting scheme. Figure 6-16 tells us that the lifting scheme is more efficient than the FFT-based DWT algorithm for lengths up to $L = 34$.

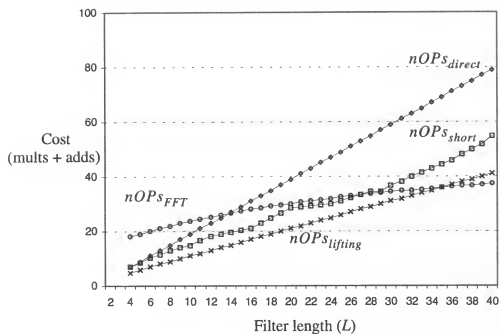


Figure 6-16. Comparison of the costs between various DWT algorithms described in Chapter 4 and the lifting scheme.

CHAPTER 7

OVERCOMPLETE LIFTING SCHEME

In this chapter we propose an *overcomplete lifting scheme* which provides a framework for multiscale analysis with overcomplete wavelet representations. As described in the previous chapter, the lifting scheme is a flexible tool for constructing compactly supported second generation wavelets which are not necessarily translates and dilates of one *mother* wavelet [29]. We have seen that the lifting scheme uses a simple basis function and builds a better performing one with desirable properties. Flexibility afforded by the lifting scheme allows basis functions to change their shapes near the boundaries without degrading regularities. The lifting scheme also provides faster processing by making optimal use of similarities between high and low pass filters. The lifting algorithm was originally introduced to construct biorthogonal wavelets associated with interpolating scaling functions. We showed in the previous chapter that the lifting scheme can be generalized with a finite number of lifting steps to construct *any* wavelets.

The lifting scheme utilizes a classical two channel filter bank as a framework for multiresolution analysis. However this traditional framework is not translation invariant [20]. Representations with a translation-invariant characteristic are highly desirable for feature analysis. Mallat and Zhong introduced an adaptive sampling based upon local maxima as an overcomplete wavelet representation [20].

In this chapter we address the following question: Can the lifting scheme be used as a framework for overcomplete wavelet representations with feature analysis in mind?

Our work addresses this question by investigating each stage of the multiscale analysis described by Sweldens [29]. We use a *smoothing Lazy* wavelet in the *split* stage which does not subsample, but smooths an input image so that the low-pass channel contains some redundant information. We then show that the following *predict* and *update* stages, based upon specific properties, indeed lead to a useful multiresolution analysis.

7.1 Undecimated Discrete Wavelet Transform with Lifting

In the previous chapter we showed how the (primal) lifting (6.19) and the dual lifting (6.21) were driven and how they can be embedded in a framework for a multiscale analysis which was implemented in the filter bank coding system shown in Figure 6-4. This computational structure was simplified later as a semantic form in Figure 6-5. Here the filters followed by successive subsampling was defined as the *Lazy* wavelet transform which simply *splits* an original signal into odd and even subsets. The dual lifting *predicted* wavelet coefficients with the help of the scaling function coefficients based upon the correlation present in the finer signal. The primal lifting then *updated* the scaling function coefficients using the previously predicted wavelet coefficients to remove possible aliasing resulted from the downsampling.

For overcomplete representations, we introduce a *smoothing lazy wavelet* in the split stage which does not downsample the input image so that $\lambda_{j+1,k} = \lambda_{j,l} = \gamma_{j,m}$. Here we choose the indices such that $j \in J$, $l \in K(j)$, $k \in K(j+1)$, and $m \in M(j)$ where $M(j) = K(j+1) \setminus K(j)$. We then smooth $\lambda_{j,k}$ using a smoothing filter, which is often a Gaussian [20]. This new splitting methodology in the filter bank scheme is then followed by the decorrelation of wavelet coefficients using the dual lifting. Since the new *split* operation does not use downsampling to make distinct subsets $\{\lambda_{i,l}\}$ and $\{\gamma_{i,l}\}$, there is

no aliasing. The primal lifting introduced to compensate for the aliasing by preserving energy between two contiguous approximations needs no longer exist. The new forward and inverse wavelet transform algorithm for overcomplete representations can thus be formulated as

FORWARD TRANSFORM

$$\begin{aligned} \text{Split: } \lambda_{j,l} &= \sqrt{2} \sum_k \tilde{h}_{k-2l} \lambda_{j+1,k}, \quad \gamma_{j,m} = \lambda_{j+1,k} \\ \text{Dual Lifting: } \gamma_{j,m} &= \gamma_{j,m} - \sum_{n=0}^{N-1} \tilde{s}_n \lambda_{j,m-2n} \end{aligned} \quad (7.1)$$

INVERSE TRANSFORM

$$\begin{aligned} \text{Inverse Dual Lifting: } \gamma_{j,m} &= \gamma_{j,m} + \sum_{n=0}^{N-1} \tilde{s}_n \lambda_{j,m-2n} \\ \text{Join: } \lambda_{j+1,k} &= \lambda_{j,l} + \gamma_{j,m} \end{aligned} \quad (7.2)$$

where $K(j+1)$, $L(j)$ and $M(j)$ are index sets such that $|K(j+1)| = |M(j)| = |K(j)|$, $\{k|k \in K(j+1)\}$, $\{l|l \in L(j)\}$ and $m|m \in M(j)$. \tilde{N} and N are primal and dual vanishing moments, respectively. Figure 7-1 shows an undecimated version of two channel analysis/synthesis filter bank with the newly proposed *overcomplete lifting scheme*. By iterating this through the low-pass channel, we can compute the 1-D discrete wavelet transform for overcomplete multiscale representations.

In the previous chapter we discussed how interpolating scaling functions are related to the lifting scheme through the interpolating subdivision proposed by Deslauriers and Dubuc [12]. Here we use the same mechanism, but the approach is different. Instead of refining samples by filling middle point between two contiguous samples at a finer

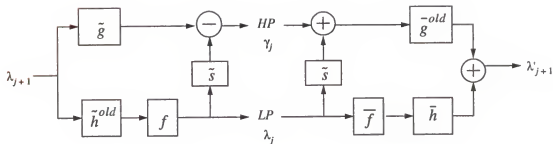


Figure 7-1. Undecimated version of two channel filter bank with the proposed overcomplete lifting.

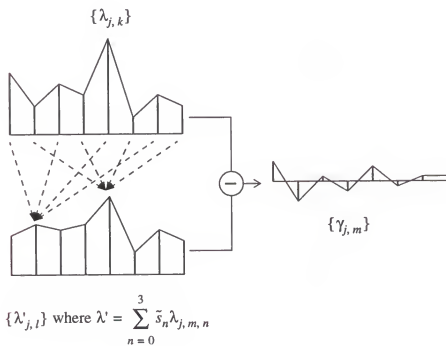


Figure 7-2. Forward wavelet transform to extract detail information as failures to be cubic.

level using subdivision with a constant spatial support, we keep the size of the samples constant throughout the decomposition, but change the support of the filter associated with the interpolating scaling function.

Figure 7-2 illustrates how to compute the wavelet coefficients with a cubic dual lifting polynomial ($N = 4$). In this figure, we show how the samples at the next analyzing level are related to the samples at the finer level using two points one of which is near the left boundary, and other not. The predicted wavelet coefficients contain the amount by which the coarser approximation locally fails to be cubic. Here the dual lifting coefficients $\tilde{s} = \{-0.0625, 0.5625, 0.5625, -0.0625\}$. The coefficients are adjusted near the boundaries of the finite input signal to avoid boundary effects. For $N = 6$, $\tilde{s} = \{0.0117, -0.0977, 0.5859, 0.5859, -0.0977, 0.0117\}$ and $\tilde{s} = \{-0.0024, 0.0239, -0.1196, 0.5981, 0.5981, -0.1196, 0.0239, -0.0024\}$ for $N = 8$.

To improve the performance of the overcomplete lifting scheme, we use the multiscale block algorithm explained in Chapter 5. In the block algorithm we keep the size of the filter kernels unchanged throughout multiscale analysis. Instead, we partition the input sequence into two sub-sequences so that each subsampled sequence contains only those pixels affected by the filter kernel. Figure 7-3 demonstrates the overcomplete lifting scheme with block algorithm applied to a sine signal with discontinuities near the middle. Figure 7-3(d) tells us that there are no boundary effects which were obviously seen in Figure 2-1(b).

For images, 2-D masks can be used. Figure 7-4(a) is a mask for $N = 4$ which is used for regions which are not affected by the boundaries. Figure 7-4(b) is for the pixel (0, 0) of an image.

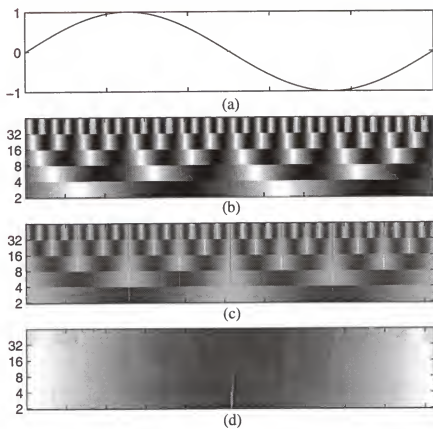


Figure 7-3. Implementation of the overcomplete lifting scheme with multiscale block algorithm (a) Original signal with discontinuities near the middle. (b) Blocked input signal. (c) Blocked wavelet coefficients after lifting. (d) Unblocked coefficients.

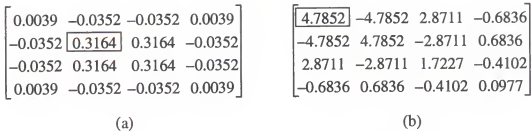


Figure 7-4. Examples of 2-D masks of dual lifting coefficients for $N = 4$. (a) A mask for regions which are not affected by the boundaries. (b) A mask for the point $(0, 0)$ of an input image. Rectangles indicate points that are modified by masking operations.

The proposed algorithm has been applied to a mammogram of size 512 by 512. We used a cubic spline-based filter for H described in [20] for the smoothing lazy wavelet transform. For the dual lifting, we applied a cubic polynomial with coefficients $\tilde{s} = \{-0.0625, 0.5625, 0.5625, -0.0625\}$. Figure 7-5 shows wavelet representations for three levels of analysis. We used the wavelet coefficients to reconstruct an image with an error on the order of 11.3dB.

7.2 Computational Cost of Overcomplete Lifting Scheme

The computation of the costs of undecimated DWT's is different from those of decimated ones. In some of the decimated DWT algorithms, filter lengths remain unchanged as the analyzing level j changes, while the input lengths are decimated by 2^{j-1} where $j = 1, 2, \dots, J$. However in undecimated DWT's the input length remains unchanged, while the filter lengths are upsampled by 2^{j-1} . In this section, we compute total costs of the various DWT's for overcomplete representations by counting all input points at each level of analysis with different input lengths. And we shall compare them to see which algorithm is most efficient.

Let j , N and L be the octave, the input length and the filter length, respectively, where $L = 2^n$. For convenience' sake, let us denote by $ODWT(X)$ an overcomplete DWT of an X algorithm. We showed in Chapter 4 that the number of operations per input point in an elementary

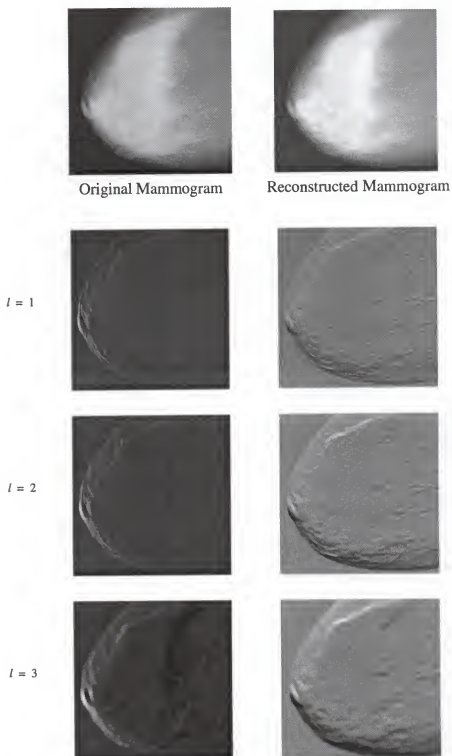


Figure 7-5. Wavelet coefficients of a mammogram with the proposed scheme. The first and second column show the wavelet coefficients in horizontal direction and vertical direction, respectively.

cell for the direct filter bank algorithm is $2L - 1$ (4.2). However the filter length L changes to $2^{j-1}(L - 1) + 1$ as the analyzing level j increases. Thus if we denote by $C_{n,L}^j(\text{direct})$ the total complexity of $ODWT(\text{direct})$ with $N = 2^n$ input points at the j th octave with length- L filters, then

$$C_{n,L}^j(\text{Direct}) = 2^n(2^j(L - 1) + 1) . \quad (7.3)$$

In the FFT-based algorithm, the filter length does not change so that $L = N = 2^n$ throughout the analysis. The cost (4.17) remains unchanged for every j . Thus the computational cost of $ODWT(\text{FFT-based})$ at the j th octave with 2^n inputs is

$$C_{n,L}^j(\text{FFT-based}) = 2^{n+1}(4n - 3) + 32 . \quad (7.4)$$

The cost of the traditional lifting scheme described in Chapter 6 was $L + 1$. Since the filter length also changes to $2^{j-1}(L - 1) + 1$ as j increases, the cost of $ODWT(\text{Lifting})$ at the octave j with 2^n inputs is thus given as

$$C_{n,L}^j(\text{Lifting}) = 2^n(2^{j-1}(L - 1) + 2) . \quad (7.5)$$

The cost of the lifting scheme (7.5) can be reduced using the block algorithm described in Chapter 5. The basic idea of the block algorithm was to keep the size of the filter kernels unchanged throughout multiscale analysis. Instead, we partition the input sequence into two sub-sequences so that each subsampled sequence contains only those pixels affected by the filter kernel. Thus for every j , the filter length is unchanged and it is L . Although we split the input of length 2^{n-j} into two subsampled inputs of length $2^{n-(j-1)}$ at the octave j , the total number of points involved at each level of analysis is unchanged since at the next level $j - 1$ the number of blocks is 2^{j-1} , and thus the total points are $2^{j-1} \cdot 2^{n-(j-1)} = 2^n$. Therefore the computational cost of $ODWT(\text{Lifting})$

with the block algorithm (i.e., $ODWT(Lifting_{block})$) at the j th octave is given as

$$C_{n,L}^j(Lifting_{block}) = 2^n(L+1). \quad (7.6)$$

In order for quantitative comparison between the complexities given by (7.3), (7.4), (7.5) and (7.6), we assign practically reasonable values to the variables, j , n and L as follows: Since multiscale analysis generally provides enough information up to the fifth octave, we have

$$j = \{x | x \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}\}.$$

A large percentage of images we process for analysis has lengths of 128, 256 and 512. Thus

$$n = \{x | x \in \{7, 8, 9\}\}.$$

Since practical applications generally use short filters,

$$L = \{x | x \in \{2, 3, 5, 7\}\}.$$

Figure 7-6 shows the computational costs of $ODWT(Direct)$, $ODWT(FFT-based)$, $ODWT(Lifting)$, and $ODWT(Lifting_{block})$ with different filter lengths $L = 2, 3, 5$ and 7 with $n = 7$. Figure 7-7 and Figure 7-8 show the same costs as Figure 7-6 with different input lengths $n = 8$ and 9 , respectively. Figure 7-9 shows the complexities with analyzing levels up to $J = 3, 4, 5$ and 6 . Here $n = 7$. Figure 7-10 and Figure 7-11 with $n = 8$ and 9 , respectively.

Since the lifting scheme can speed up the wavelet transform by a factor of two and convolution using a *block algorithm* can further reduces processing time by two fold within each analyzing level, the proposed algorithm can achieve an overall speedup of 4 over the standard overcomplete wavelet transform where an FFT is the basic implementation tool.

- ◇— $ODWT(Direct)$
- $ODWT(FFT\text{-}based)$
- △— $ODWT(Lifting)$
- $ODWT(Lifting_{block})$

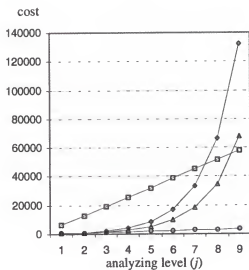
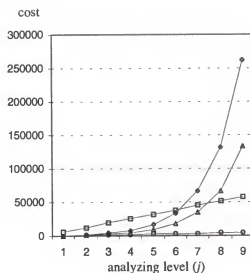
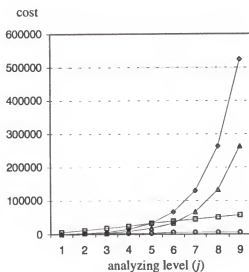
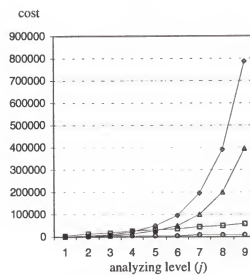
(a) $L = 2$ (b) $L = 3$ (c) $L = 5$ (d) $L = 7$

Figure 7-6. Costs of $ODWT(Direct)$, $ODWT(FFT\text{-}based)$, $ODWT(Lifting)$ and $ODWT(Lifting_{block})$ algorithms with filter length (a) $L = 2$, (b) $L = 3$, (c) $L = 5$, and (d) $L = 7$. Here $n = 7$ ($N = 128$).

- ◇ $ODWT(Direct)$
- $ODWT(FFT-based)$
- △ $ODWT(Lifting)$
- $ODWT(Lifting_{block})$

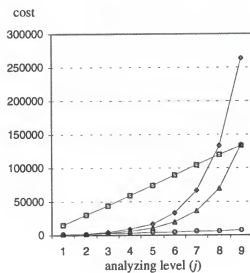
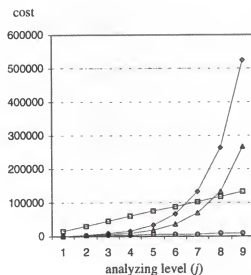
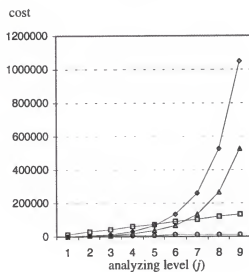
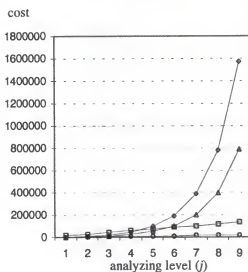
(a) $L = 2$ (b) $L = 3$ (c) $L = 5$ (d) $L = 7$

Figure 7-7. Same as Figure 7-6 with $n = 8$ ($N = 256$).

- \diamond ODWT(Direct)
 \square ODWT(FFT-based)
 \triangle ODWT(Lifting)
 \circ ODWT(Lifting_{block})

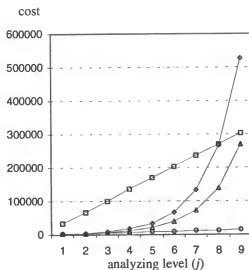
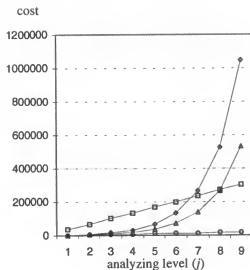
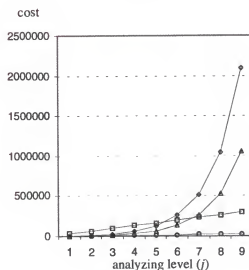
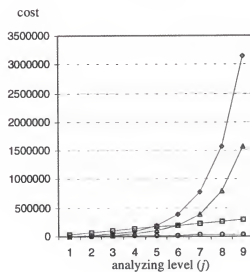
(a) $L = 2$ (b) $L = 3$ (c) $L = 5$ (d) $L = 7$

Figure 7-8. Same as Figure 7-6 with $n = 9$ ($N = 512$).

- ◇ $ODWT(Direct)$
- $ODWT(FFT-based)$
- △ $ODWT(Lifting)$
- $ODWT(Lifting_{block})$

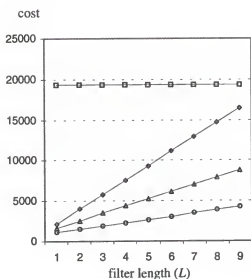
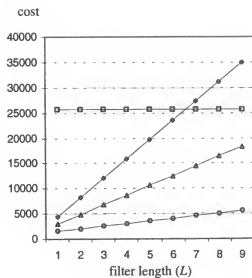
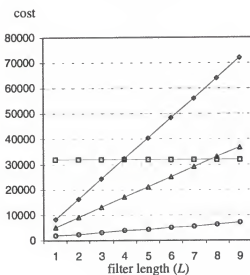
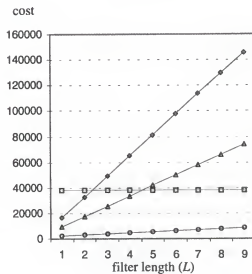
(a) $J = 3$ (b) $J = 4$ (c) $J = 5$ (d) $J = 6$

Figure 7-9. Costs of $ODWT(direct)$, $ODWT(FFT-based)$, $ODWT(lifting)$ and $ODWT(lifting_{block})$ algorithms up to analyzing levels (a) $J = 3$, (b) $J = 4$, (b) $J = 5$, and (d) $J = 6$. Here $n = 7$ ($N = 128$).

- ◇ $ODWT(Direct)$
- $ODWT(FFT-based)$
- △ $ODWT(Lifting)$
- $ODWT(Lifting_{block})$

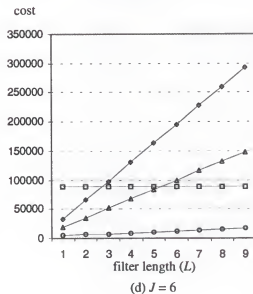
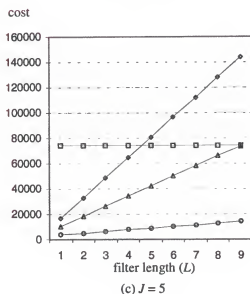
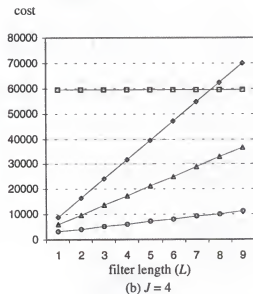
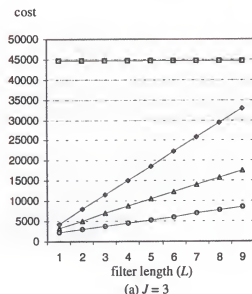


Figure 7-10. Same as Figure 7-9 with $n = 8$ ($N = 256$).

- ◇— $ODWT(Direct)$
- $ODWT(FFT-based)$
- △— $ODWT(Lifting)$
- $ODWT(Lifting_{block})$

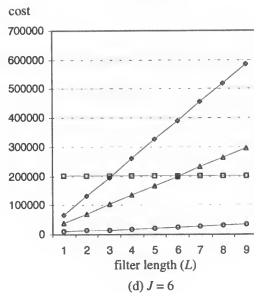
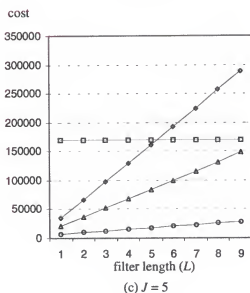
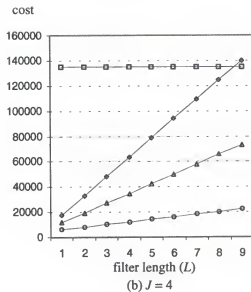
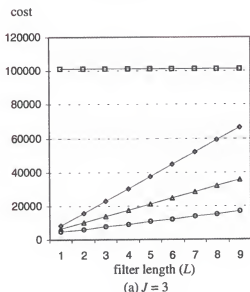


Figure 7-11. Same as Figure 7-9 with $n = 8$ ($N = 512$).

Based on the cost analysis we made in this section, we now conclude that our approach for computing overcomplete wavelet representations with compactly supported lifting filters $ODWT(Lifting_{block})$, compared to existing algorithms, is significantly more efficient.

CHAPTER 8

CONCLUSIONS

In this dissertation we showed that the lifting scheme can be used for overcomplete multiscale wavelet representations with better performance without introducing boundary artifacts that exists in the traditional processing methods. Combined with the multiscale block algorithm with support from both the hardware (SX frame) and the software (XIL imaging library), the new proposed lifting scheme helped an overcomplete wavelet transform to achieve an overall speedup of four over the traditional methods where an FFT is the basic implementation tool. We showed that the significantly improved performance of the proposed method makes undecimated wavelet transforms for feature analysis very viable in interactive processing paradigms such as a Web-based client-server model where the fast processing is highly desirable.

We also illustrated that the basis functions used in our approach are well adapted to the boundary of a finite-length input, so that the boundary distortions introduced by signal extensions do not appear. This property is very useful in some applications on general domains and in analyzing data on curves and surfaces. We also showed that the inverse transform led to a perfect reconstruction of the original input.

REFERENCES

- [1] D. Benn, J. Pettigrew, M. Shim, A. Laine, and A. Mancuso, "Automated image recognition of facial bones," *Computer Assisted Radiology*, France, Paris: Springer, pp. 943-948, 1996.
- [2] D. Benn, M. Shim, C. DeBose, A. Laine, A. Mancuso, J. Pettigrew, and H. Stambuk, "A pilot study of automated knowledge-based recognition of facial bones from axial CT images," *Proc. Int. Symp. on Comp. and Comm. Sys. for Image Guided Diagnosis and Therapy*, Germany, Berlin: Springer, pp. 1021-1027, 1995.
- [3] C. Chang, and A. Laine, "Enhancement of Mammograms from Oriented Information," To appear in *Proc. IEEE Int. Conf. Image Process.*, Santa Barbara, CA, Oct. 1997.
- [4] C. K. Chui, *An Introduction to Wavelets*, Academic Press, San Diego, CA, 1992.
- [5] C. K. Chui, ed., *Wavelets: A Tutorial in Theory and Applications*, Academic Press, San Diego, CA, 1992.
- [6] A. Cohen, "Biorthogonal Wavelets," in *Wavelets: A Tutorial in Theory and Applications*, C. K. Chui, ed., Academic Press, San Diego, CA, 1992, pp. 123-152.
- [7] A. Cohen, I. Daubechies, and J. Feauveau, "Biorthogonal Bases of Compactly Supported Wavelets," *Comm. Pure Appl. Math.*, Vol 45, pp. 485-560, 1992.
- [8] J. M. Combes, A. Grossman, and Ph. Tchamitchian, editors, "Wavelets: Time-Frequency Methods and Phase Space," *Inverse problems and theoretical imaging*, Springer-Verlag, New York, 1989.
- [9] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Comm. Pure Appl. Math.*, Vol 41, pp. 909-996, 1988.
- [10] I. Daubechies, *Ten Lectures on Wavelets*, Philadelphia, PA:SIAM, 1992.
- [11] I. Daubechies and W. Sweldens, "Factoring Wavelet Transforms into Lifting Steps," *Preprint*, Bell Laboratories, Lucent Technologies, 1996.
- [12] G. Deslauriers, and S. Dubuc, "Symmetric iterative interpolating processes," *Constr. Approx.*, Vol 5, No. 1, pp. 49-68, 1989.
- [13] D.L. Donoho, "Interpolating wavelet transforms," *Preprint*, Dept. Statistics,

Stanford University, 1992.

- [14] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution Analysis of Arbitrary Meshes," *SIGGRAPH 95 Conference Proceedings*, ACM SIGGRAPH, Addison-Wesley, pp. 173-182, 1995.
- [15] B. Jawerth and W. Sweldens, "An overview of wavelet based multiresolution analysis," *SIAM Rev.*, Vol.36, nr.3, pp.377-412, 1994.
- [16] I. Koren, A. Laine, F. Taylor, and M. Lewis, "Interactive Wavelet Processing and Techniques Applied to Digital Mammography," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, vol. 3, pp. 1415-1418, Atlanta, GA, May 1996.
- [17] A. Laine, S. Schuler, J. Fan, and W. Huda, "Mammographic feature enhancement by multiscale analysis," *IEEE Trans. Med. Imaging*, Vol. 13, No. 14, pp. 725-740, 1994.
- [18] M. Lounsbery, T. DeRose, and J. Warren, "Multiresolution Analysis for Surfaces of Arbitrary Topological Type," *ACM Trans. on Graphics*, Vol. 16, No. 1, pp 34-73, 1997.
- [19] S. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 7, pp. 674-693, 1989.
- [20] S. Mallat and S. Zhong, "Characterization of signals from multiscale edges," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 14, pp. 710-732, 1992.
- [21] Z. J. Mou and P. Duhamel, "Short-Length FIR Filters and Their Use in Fast Nonrecursive Filtering," *IEEE Trans. on Signal Processing*, Vol. 39, pp.1322-1332, 1991.
- [22] P. Duhamel, "Implementation of Split-Radix FFT Algorithms for Complex, Real, and Real-Symmetric Data," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 2, pp. 285-295, 1986.
- [23] O. Rioul and P. Duhamel, "Fast Algorithms for Discrete and Continuous Wavelet Transforms," *IEEE Trans. on Inform. Theory*, Vol. 38, No. 2, pp. 569-586, 1992.
- [24] M. J. Shensa. "Wedding the \hat{a} trous and Mallat algorithm," *IEEE Trans. Signal Process.*, Vol 40, No. 10, pp. 2464-2482, 1992.
- [25] M. Shim and A. Laine, "A fast algorithm to support interactive wavelet processing on a radiologist workstation," *Tech. Report TR97-022*, Dept. of Computer and Information Sciences and Engineering, University of Florida, 1997
- [26] G. Strang and T. Nguyen, editors, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, MA, 1996.
- [27] Sun Microsystems, Inc., "SPARCstation 10SX Graphics Technology," *White*

Paper, October, 1993.

- [28] Sun Microsystems, Inc., "XIL Imaging Library," *White Paper*, 1994.
- [29] W. Sweldens, "The Lifting Scheme: A New Philosophy in Biorthogonal Wavelet Constructions," *Wavelet Applications III*, A. Laine and M. Unser, Ed., Proc. SPIE, San Diego, CA, Vol 2569, pp. 68-79, 1995.
- [30] W. Sweldens, "The Lifting Scheme: A custom-design construction of biorthogonal wavelets," *Appl. Comput. Harmon. Anal.*, Vol. 3, No. 2, pp. 186-200, 1996.
- [31] W. Sweldens, "Building your own wavelets at home," In *Wavelets in Computer Graphics*, ACM SIGGRAPH Course notes, pp. 15-87, 1996.
- [32] M. Vetterli and C. Herley, "Wavelets and filter bank: theory and design," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 40, No. 9, pp. 2207-2232, 1992.

BIOGRAPHICAL SKETCH

Minbo Shim was born in Seoul, Korea, in 1960. He received his BSME degree from Seoul National University in 1987.

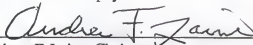
He started his master's program in the Computer and Information Science and Engineering department at the University of Florida in 1991. During this program, he developed a fast parallel stereo matching algorithm based on multiscale edges.

He shifted to the Ph.D. program in 1993, and was involved in numerous projects in the College of Dentistry. Main interests were in segmentation of objects from a set of axial CT images, and 3D volume reconstruction of anatomical structures of interest. During those years, two PC-based programs were developed and sold to US Navy and Hamburg University in Germany. The University of Florida holds the copyright of these programs.

He has been the president of the Korean Student Association at the University of Florida since 1997. He has been nominated in Outstanding Student Volunteer Award of the University of Florida.

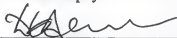
His research interests include digital signal and image processing, wavelet analysis, real-time processing, stereo imaging, medical imaging, computer and machine vision, 3D processing and computer graphics.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and in fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



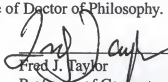
Andrew F. Laine, Chairman
Associate Professor of Computer and
Information Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and in fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



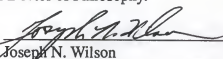
Douglas K. Benn, Cochairman
Associate Professor of Dental Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and in fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



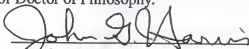
Fred J. Taylor
Professor of Computer and Information
Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and in fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Joseph N. Wilson
Assistant Professor of Computer and
Information Science and Engineering


I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and in fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



John G. Harris
Assistant Professor of Electrical and Computer
Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1998



Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School